



United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

CSE 4889: Machine Learning: Section: A

Total Marks: 40 Duration: 2.00 hrs

There are 10 questions. Answer 8 questions. Every question carries equal marks

****Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.**

1. You know that XOR function cannot be implemented by a single layer neural network. Show in a step by step fashion that XOR function can be implemented by a multilayer neural network
2. Assume that you have a two input neural network. You have three hidden neurons in the first hidden layer, one output neuron in the output layer and you are using a step function as the activation function. Show geometrically that the input data is mapped into vertices of a three dimensional unit length cube by the hidden layer neurons. What is the function of the output neuron?
3. What steps can you take to avoid local minima during training of a neural network? How does momentum term increase the convergence speed of training a neural network?
4. What is cross entropy function? When is it minimum and when is it maximum? What has its advantage over other cost functions? What should be the activation function of the output nodes if you use cross entropy cost function and why?
5. How can you control the size of a neural network using parameter sensitivity calculation and cost function regularization

6. From the definition of support vector machine formulate the Lagrangian optimization problem. Now derive the following where the symbols carry their usual meaning:

$$\mathbf{w} = \sum_{t=1}^N \lambda_t y_t \mathbf{x}_t$$

$$\sum_{t=1}^N \lambda_t y_t = 0$$

Which examples are the support vectors? What are the values of Lamdas for support vectors? For which training examples the Lamda values are zeros?

7. Three men are tossing coins with the probabilities of coming up head and tail are $\langle 0.4, 0.6 \rangle$, $\langle 0.7, 0.3 \rangle$ and $\langle 0.5, 0.5 \rangle$ respectively. If the tossing is done by the 1st man, the probability of tossing next by the 1st, 2nd and 3rd man are 0.4, 0.3 and 0.3 respectively. If the tossing is done by the 2nd man, the probability of tossing next by the 1st, 2nd and 3rd man are 0.5, 0.2 and 0.3 respectively. If the tossing is done by the 3rd man, the probability of tossing next by the 1st, 2nd and 3rd man are 0.4, 0.2 and 0.4 respectively. The three men tossed four times and the outcome was H, T, H, H. Assume that the probability of tossing will be started by 1st, 2nd or 3rd man are 0.4, 0.3 and 0.3 respectively. Find the optimal state sequence to make the probability of the given outcome maximum.
8. An input image $227 \times 227 \times 3$ is convolved with 100 filters having size $= 21 \times 21 \times 3$, stride = 4 and zero padding = 1. In the 2nd phase output is convolved with 50 filters having size $= 7 \times 7 \times 100$, stride = 2 and zero padding = 0. In the third phase the output is convolved with 10 filters having size $= 5 \times 5 \times 50$, stride = 1 and zero padding = 0. Find out the sizes of the outputs after each phase. What will be the total number of neurons needed?
9. In Question #8, show the implementation of the 1st phase using matrix multiplication. If you want to avoid pooling for downsizing the output, what alternative can you use?
10. How can you calculate the probability of a sequence of observation under Hidden Markov Model using forward procedure? What is its complexity?

Student Id: 011201205

1. In a two input neural network there are three neurons in the first hidden layer and one neuron in the output layer. Show the polyhedral regions formed by the hidden units and mark the regions which map into 110 and 111. What is the dimension of the plane produced by the output unit?

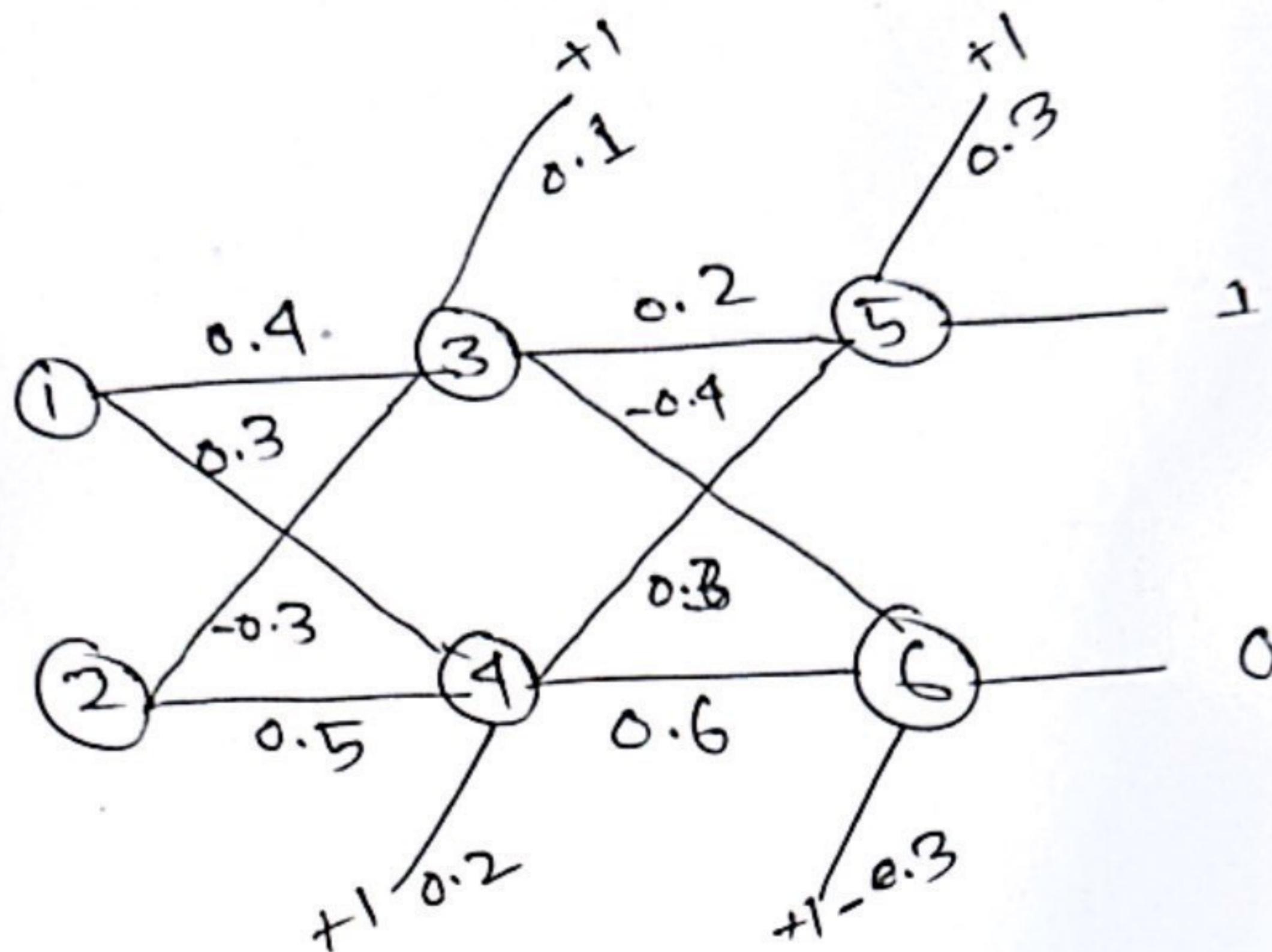
2. There are two output units in a multilayer neural network. For some input example the outputs are 0.8 and 0.2 respectively where the targets are 1.0 and 0.0 respectively. For a 2nd example the outputs are 0.6 and 0.4 respectively where the targets are 0.0 and 1.0 respectively. Find the value of the cross entropy cost function for these two examples.

Mid

Q1

$$x_1 = 1$$

$$x_2 = 1$$



I_j

$$\leq (\text{input}_j \times w_j)$$

$$\begin{aligned}
 I_3 &= x_1 \times w_{13} + x_2 \times w_{23} + \cancel{x_1} \times w_{03} \\
 &= 1 \times 0.4 + 1 \times (-0.3) + 1 \times 0.1 \\
 &= 0.2
 \end{aligned}$$

$$\begin{aligned}
 I_4 &= x_1 \times w_{14} + x_2 \times w_{24} + 1 \times w_{04} \\
 &= 1 \times 0.3 + 1 \times 0.5 + 1 \times 0.2 \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 I_5 &= o_3 \times w_{35} + o_4 \times w_{45} + 1 \times w_{05} \\
 &= 0.55 \times 0.2 + 0.73 \times 0.3 + 1 \times 0.3 \\
 &= 0.629
 \end{aligned}$$

$$\begin{aligned}
 I_6 &= o_3 \times w_{36} + o_4 \times w_{46} + 1 \times w_{06} \\
 &= 0.55 \times (-0.4) + 0.73 \times 0.6 + 1 \times (-0.3) \\
 &= -0.082
 \end{aligned}$$

$$o_j \leftarrow \frac{1}{1 + e^{-I_j}}$$

$$o_3 = \frac{1}{1 + e^{-0.2}} = 0.55$$

$$o_4 = \frac{1}{1 + e^{-1}} = 0.73$$

$$o_5 = \frac{1}{1 + e^{-0.629}} = 0.65$$

$$o_6 = \frac{1}{1 + e^{(-0.082)}} = 0.48$$

$$\Delta_j = Err_j \times o_j \times (1-o_j)$$

$$\begin{aligned}\Delta_5 &= (target - o_5) \times o_5 \times (1-o_5) \\ &= (1 - 0.5) \times 0.5 \times (1-0.5) \\ &= (1 - 0.65) \times 0.65 \times (1-0.65) \\ &= 0.08\end{aligned}$$

$$\begin{aligned}\Delta_6 &= (0 - 0.48) \times 0.48 \times (1-0.48) \\ &= -0.12\end{aligned}$$

$$\begin{aligned}\Delta_3 &= Err_3 \times o_3 \times (1-o_3) \\ &= (\Delta_5 \times w_{35} + \Delta_6 \times w_{36}) \times o_3 \times (1-o_3) \\ &= \{0.08 \times 0.2 + (-0.12) \times (-0.4)\} \times 0.55 \times (1-0.55) \\ &= 0.016\end{aligned}$$

$$\begin{aligned}\Delta_4 &= Err_4 \times o_4 \times (1-o_4) \\ &= (\Delta_5 \times w_{45} + \Delta_6 \times w_{46}) \times o_4 \times (1-o_4) \\ &= \{0.08 \times 0.3 + (-0.12) \times 0.6\} \times 0.73 \times (1-0.73) \\ &= -0.0095\end{aligned}$$

change the weight of every links

$$w_{ij} = w_{ij}(\text{old}) + \alpha \times 0.1 \times \Delta_6$$

$$\begin{aligned} w_{13} &= w_{13}(\text{old}) + \alpha \times 0.1 \times \Delta_3 \\ &= 0.4 + 0.9 \times 1 \times 0.016 \\ &= 0.414 \end{aligned}$$

$$\begin{aligned} w_{14} &= w_{14}(\text{old}) + \alpha \times 0.1 \times \Delta_4 \\ &= 0.3 + 0.9 \times 1 \times (-0.0095) \\ &= 0.299 \end{aligned}$$

$$\begin{aligned} w_{23} &= -0.3 + 0.9 \times 1 \times 0.016 \\ &= -0.29 \end{aligned}$$

$$\begin{aligned} w_{24} &= 0.5 + 0.9 \times 1 \times (0.0095) \\ &= 0.49 \end{aligned}$$

$$\begin{aligned} w_{35} &= 0.2 + 0.9 \times 0.55 \times (0.03) \\ &= 0.24 \end{aligned}$$

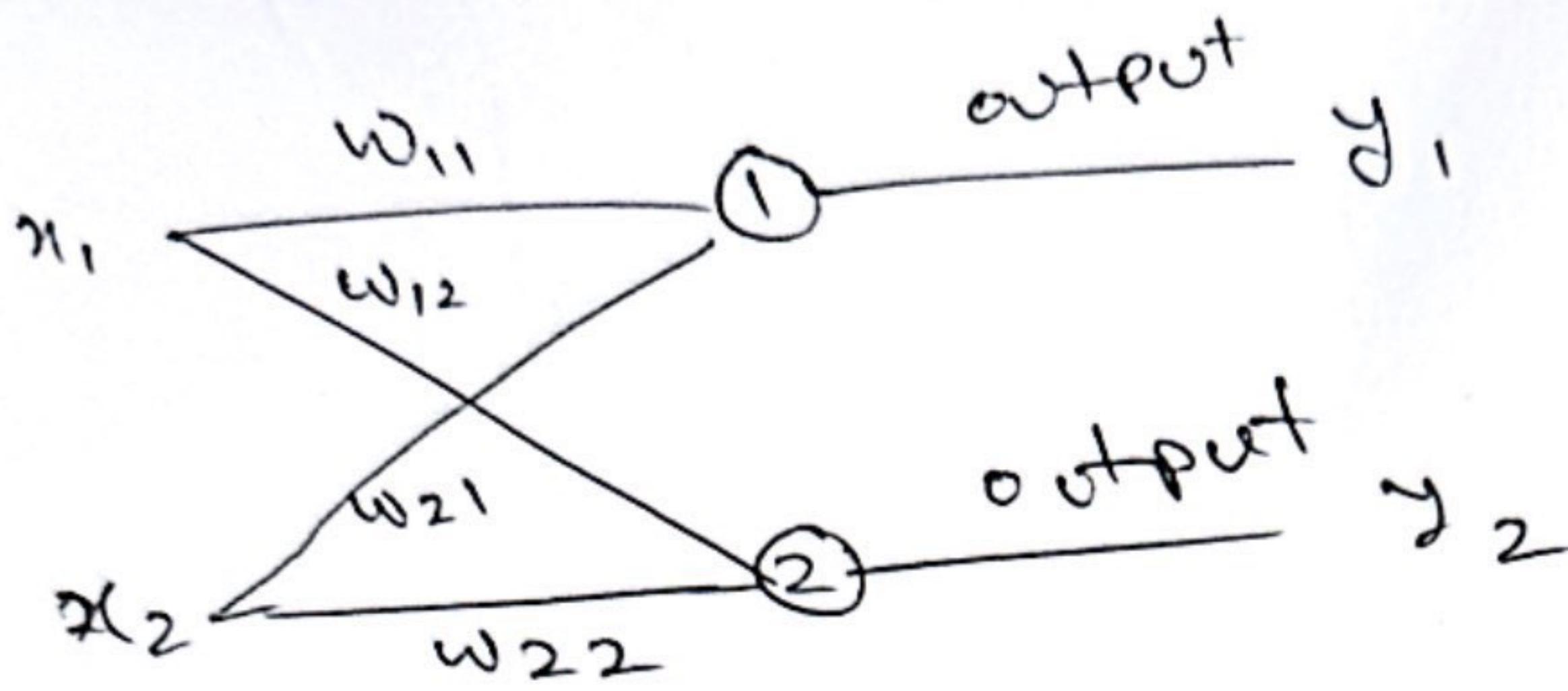
$$\begin{aligned} w_{36} &= -0.4 + 0.9 \times 0.55 \times (-0.12) \\ &= -0.46 \end{aligned}$$

$$w_{45} = 0.3 + 0.9 \times 0.73 \times 0.02 \\ = 0.35$$

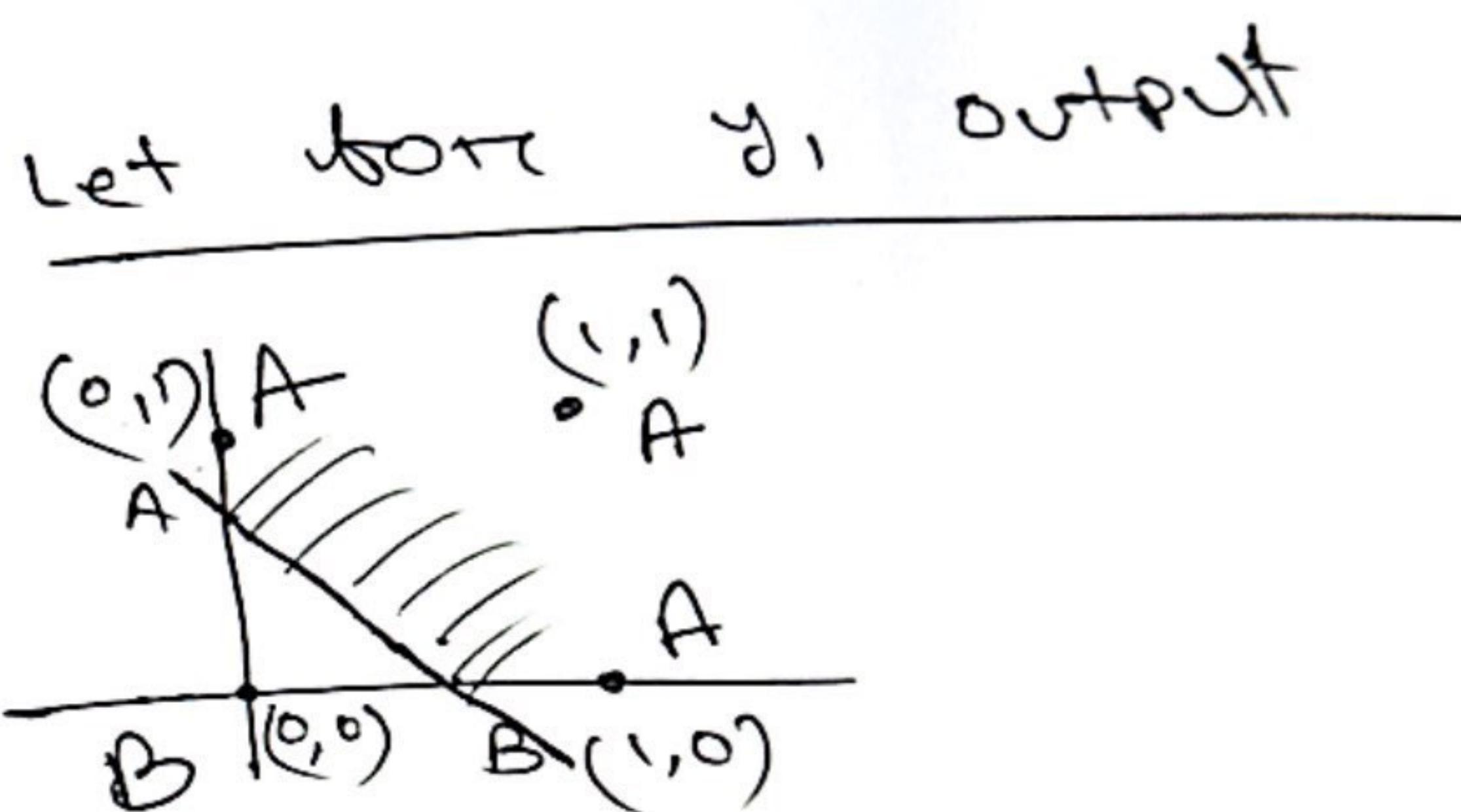
$$w_{46} = 0.6 + 0.9 \times 0.73 \times (-0.12) \\ = 0.52$$

21

■ XOR function by multilayer neural network:



x_1	x_2
0	0
0	1
1	0
1	1

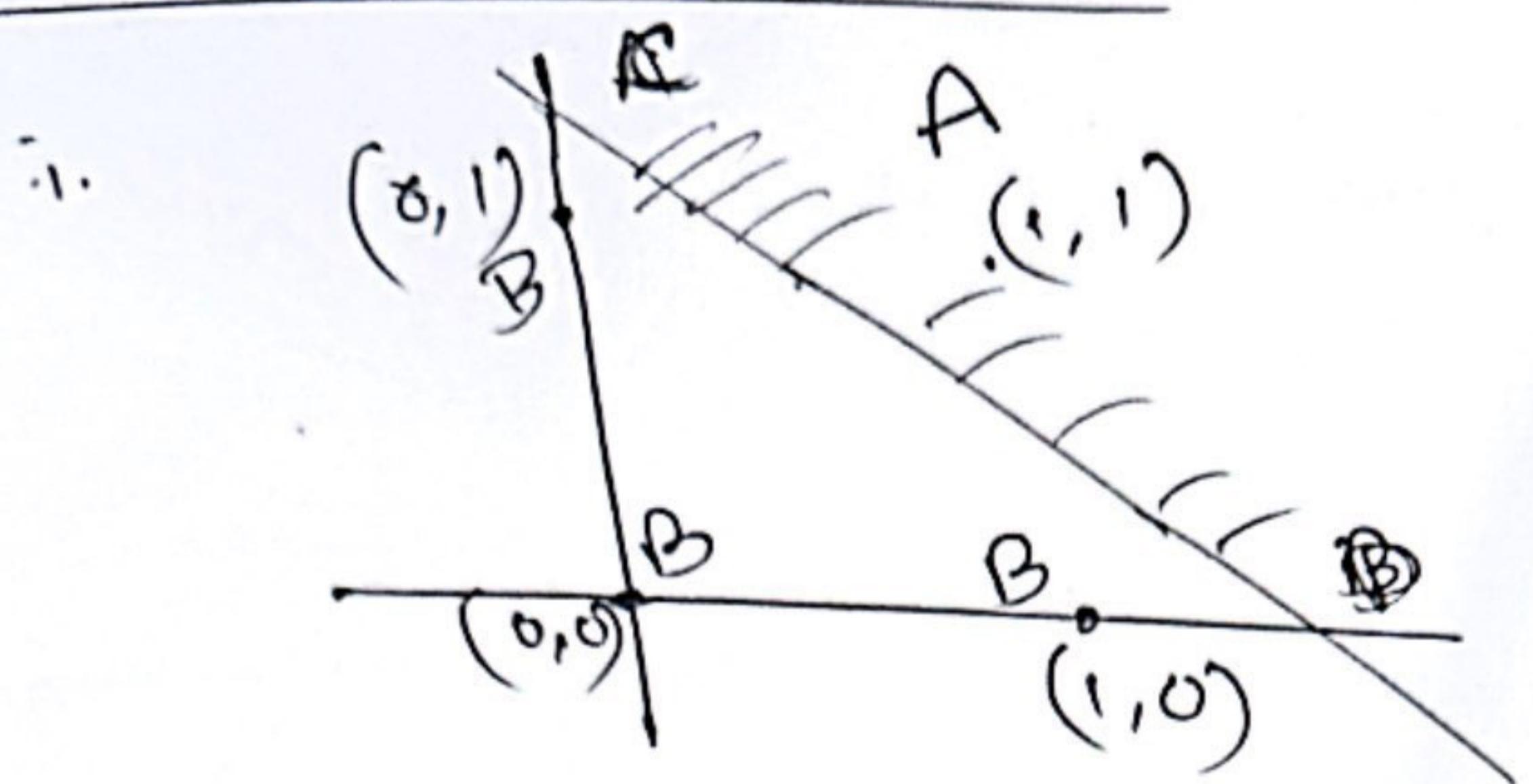


if for y_1 , threshold becomes
AB line then the output
will be 0, 1, 1, 1

x_1	x_2	y_1
0	0	0
0	1	1
1	0	1
1	1	1

link OR gate

for y_2 output

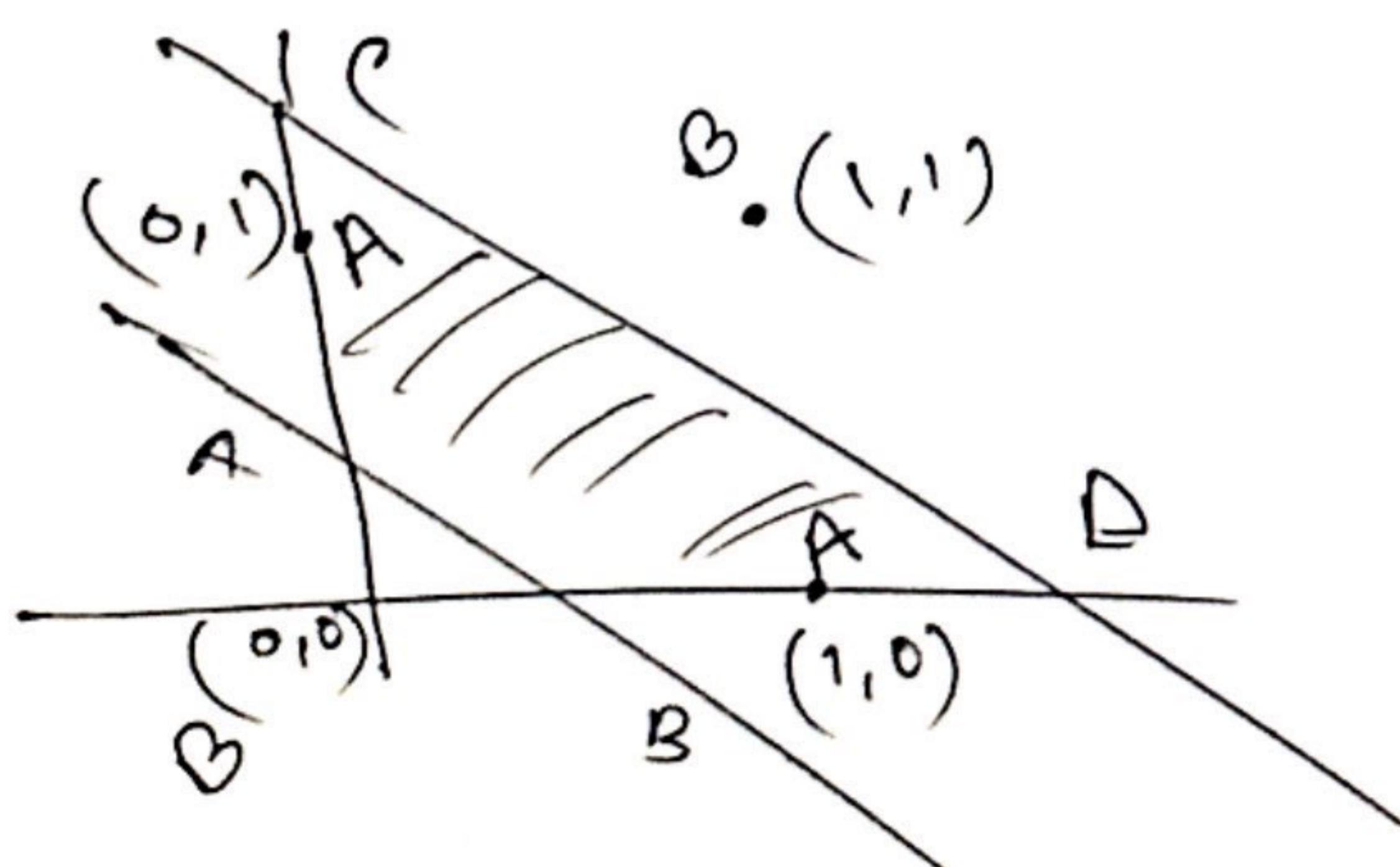


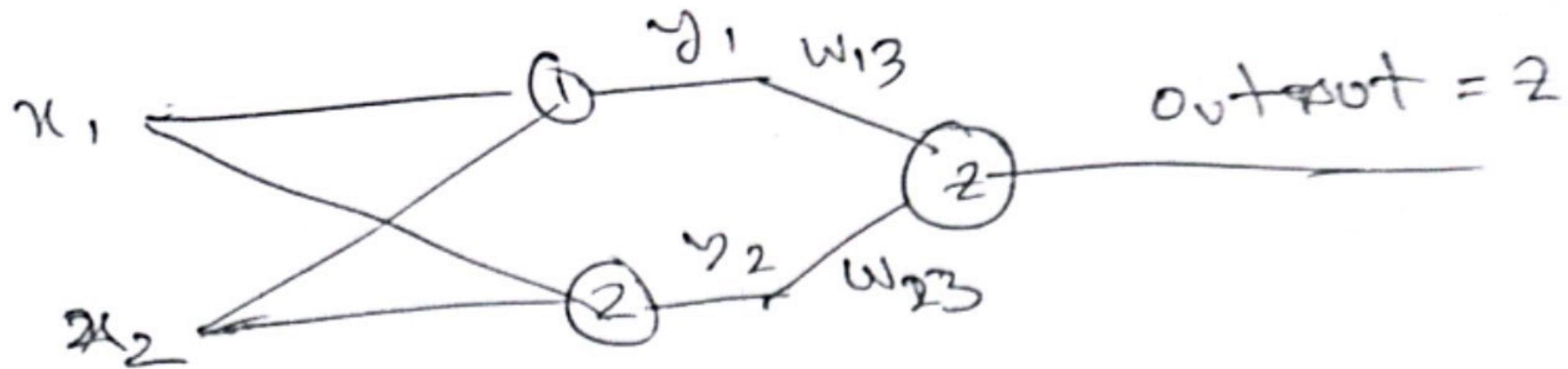
if for y_2 threshold becomes
CD line then the output
will be 0, 0, 0, 1

x_1	x_2	y_2
0	0	0
0	1	0
1	0	0
1	1	1

like AND gate

Now, if set the threshold a range
between AB and CD then the
output will be change



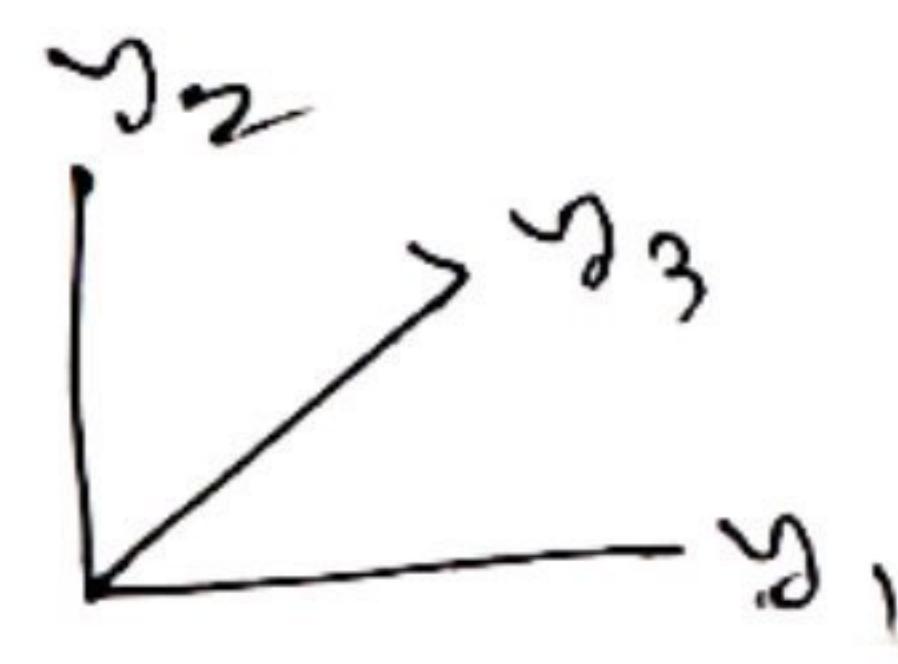
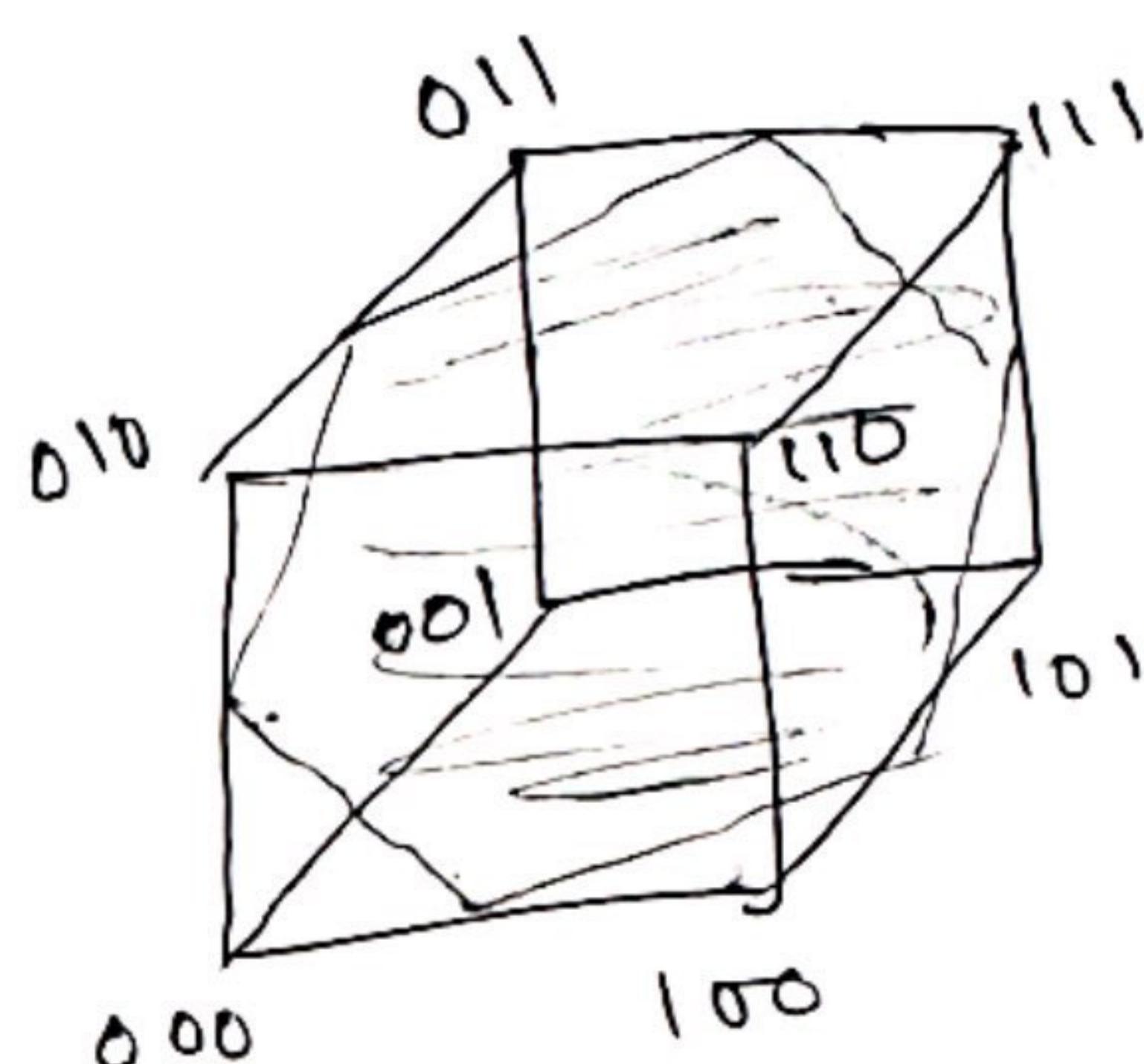
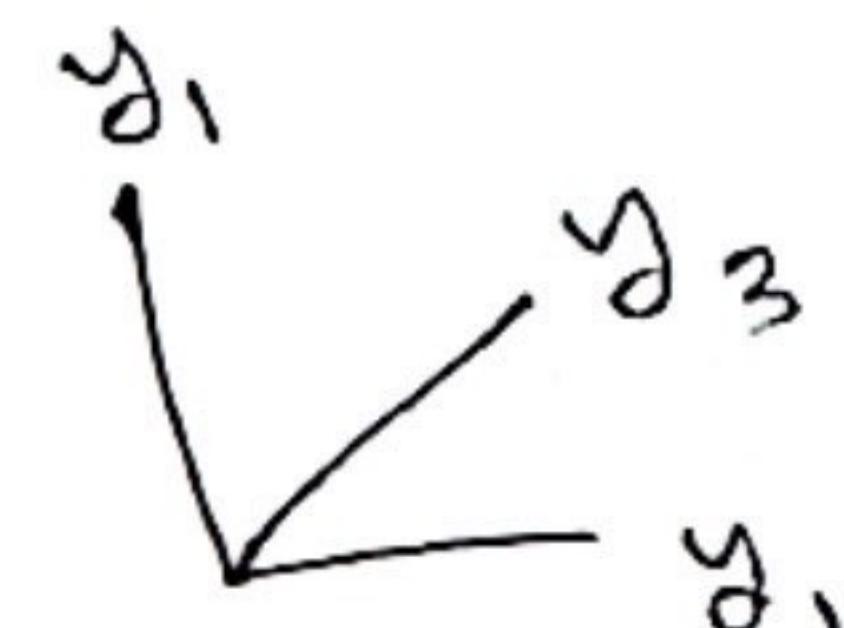
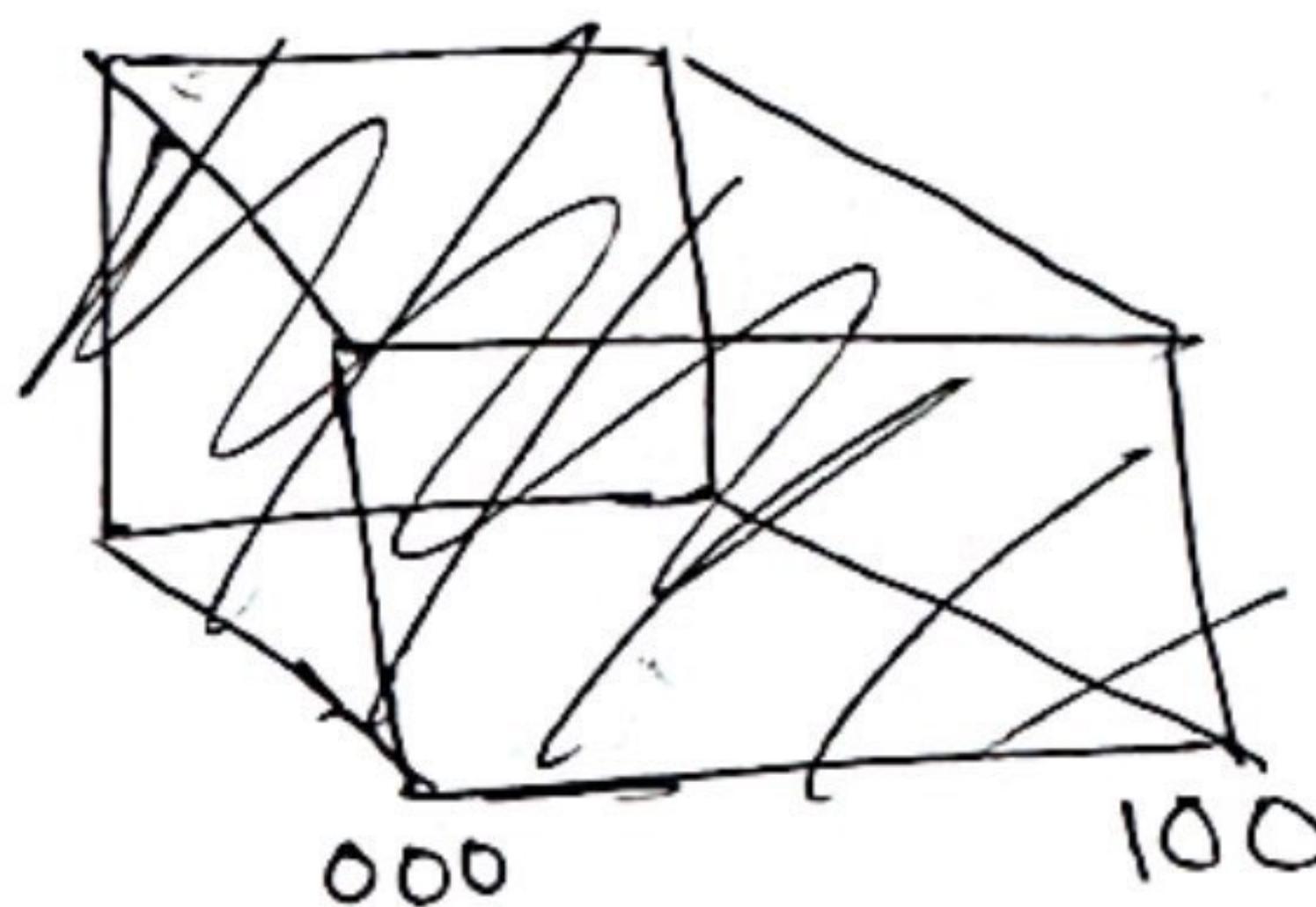


x_1	x_2	z
0	0	0
0	1	1
1	0	1
1	1	0

So the z function act like
XOR function ~~and~~ when y_1 and
 y_2 are the input of z .

This is the way XOR function
implemented by multilayer neural
network.

21. for hidden layer 3 input ^{neuron} and one output neuron, the geometrical mapping at vertices of a three dimensional unit length cube:



The output realizes a hyper plane to separate vertices according to their class label.

Q) The function of output neuron:

Ans: To perform a binary decision or classification based. (0 or 1) or (A or B) using step function as activation function.

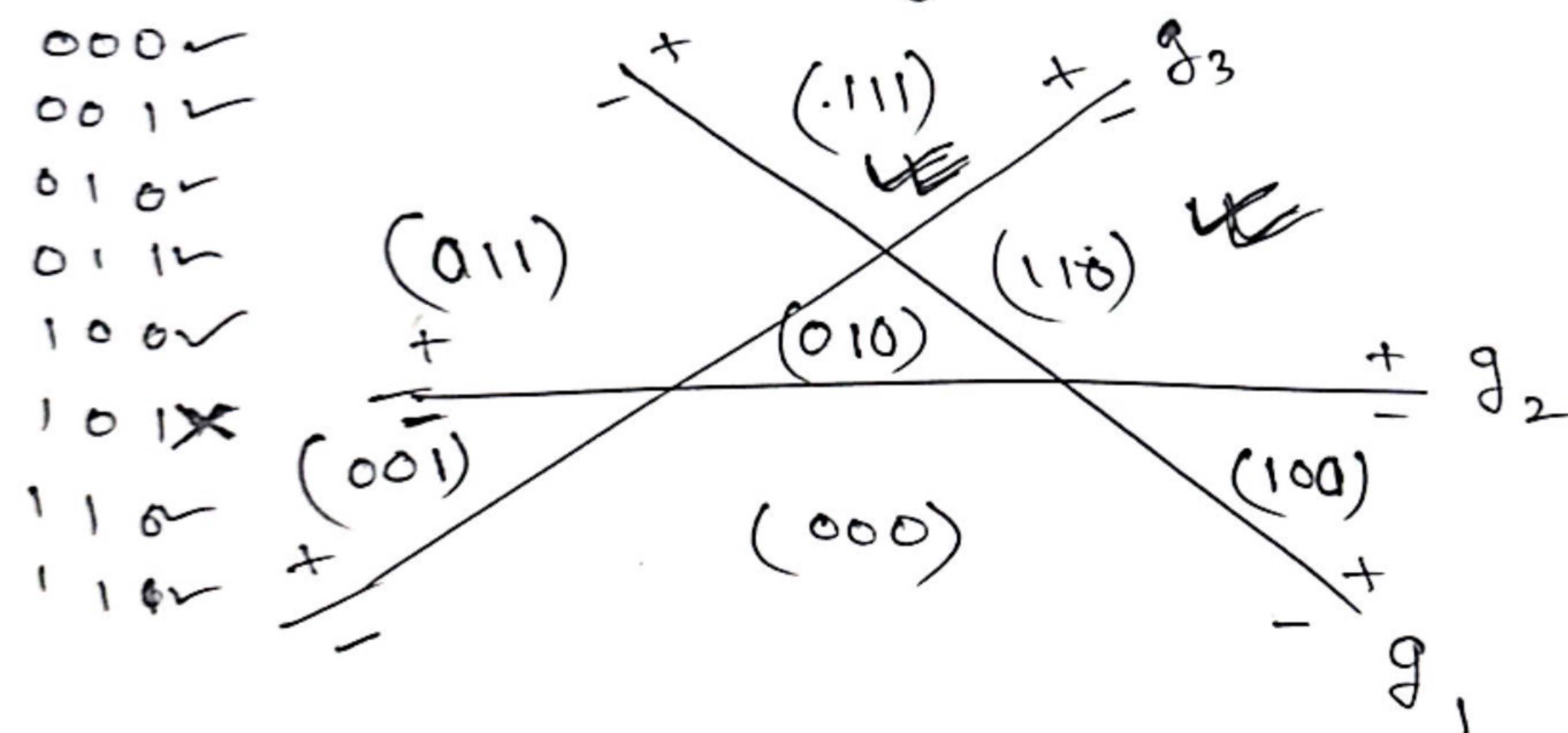
CT

1) Given -

three neurons in the first hidden layer. And one output neuron.

Polyhedral regions:

g_1, g_2, g_3 as 3 neuron
(+) and (-) determines the input 1 and 0 respectively.



what is + [Note: 101 vertex does not correspond to any regions of output polyhedral. It means, it is dimension of output unit:
as a virtual polyhedra. This will not influence the classification task.]

As Hidden layer dimension is 3 (three input)
So, output hyper plane dimension is: 2

31 To avoid local minima:

- 1) ~~startin~~ Random Restart (Hill climbing):
it can be started from a random position and update the weight and fix the error. The process to be continued.
- 2) Batch mode: After iterating all the example and summation of the error it should update the weight and fix it.
- 3) Pattern mode: After iterating example it fixes every single the error and update the weight.

For momentum term increase

$$\Delta w_{jk}(\text{new}) = \alpha \Delta w_{jk}(\text{old}) + \eta \alpha \Delta_k$$
$$w_{jk}(\text{new}) = w_{jk}(\text{old}) + \Delta w_{jk}(\text{new})$$

$$\therefore \Delta w_{jk}(t) = \alpha \Delta w_{jk}(t-1) + \eta g(t)$$

$$\Delta w_{jk}(1) = \alpha \Delta w_{jk}(0) + \eta g(1)$$

$$\Delta w_{jk}(2) = \alpha \Delta w_{jk}(1) + \eta g(2)$$

$$= \alpha (\alpha \Delta w_{jk}(0) + \eta g(1)) + \eta g(2)$$
$$= \alpha^2 \Delta w_{jk}(0) + \alpha \eta g(1) + \eta g(2)$$

$$\Delta w_{jk}(3) = \alpha \Delta w_{jk}(2) + \eta g(3)$$

$$= \alpha (\alpha^2 \Delta w_{jk}(0) + \alpha \eta g(1) + \eta g(2))$$
$$+ \eta g(3)$$

$$= \alpha^3 \Delta w_{jk}(0) + \alpha^2 \eta g(1) + \alpha \eta g(2)$$
$$+ \eta g(3)$$

$$= \alpha^3 \Delta w_{jk}(0) + \eta (\alpha^2 g(1) + \alpha^1 \eta g(2) + \alpha^0 g(3))$$

$$\Delta w_{jk}(T) = \alpha^T \Delta w_{jk}(0) + \gamma (\alpha^{T-1} g(1) + \alpha^{T-2} g(2) + \dots + g(T))$$

$$\Delta w_{jk}(T) = \underbrace{\alpha^T \Delta w_{jk}(0)}_0 + \gamma \sum_{t=0}^{T-1} \alpha^t g(T-t)$$

$$= \eta (1 + \alpha + \alpha^2 + \dots + \alpha^{T-1}) g$$

$$= \eta \frac{1}{1-\alpha} \times g$$

$$= \frac{n}{1-\alpha} \times g \quad [\text{where } g \text{ is gradient which is very small}]$$

So, momentum could be increase by increasing the value of η .

$$\text{As } \Delta w_{jk}(T) \propto \eta$$

and η is the controller of the contribution of the momentum term to the update direction.

4)

Cross entropy function:

- The cross-entropy function is a commonly used cost function in machine learning particularly in classification tasks. It measures the difference between the predicted probability distribution and the true probability distribution at the target variable.
- And it also a well formed function which doesn't trap at local minima.

$$T = \sum_{i=1}^N \sum_{k=1}^K \left(y_k^{(i)} \ln \hat{y}_k^{(i)} + (1 - y_k^{(i)}) \ln (1 - \hat{y}_k^{(i)}) \right)$$

T is the cross entropy cost function

where $y_k^{(i)}$ is actual output

$\hat{y}_k^{(i)}$ is predicted output

minimum T if $y_k(i) = \hat{y}_k^A(i)$

maximum T if $y_k(i) \neq \hat{y}_k^A(i)$ and
means completely different

The activation function could be
sigmoid function for the output
node.

Because: Sigmoid function maps the
output at the last layer
at the neural network to
a value between 0 and 1.

CT - section (B)

2) 1st example:

output : 0.8, 0.2,

target : 1.0, 0.0

$\therefore y_k^{(i)} = 0.8, 0.2$ (actual)

$\hat{y}_k^{(i)} = 0.8, 0.2$ (prediction)

Cross-entropy cost function

$$L = \sum \sum \left[y_k^{(i)} \ln \hat{y}_k^{(i)} + (1-y_k^{(i)}) \ln (1-\hat{y}_k^{(i)}) \right]$$

$$= \sum \sum y \ln y' + (1-y) \ln (1-y')$$

$$= \left[y_1 \ln y'_1 + (1-y_1) \ln (1-y'_1) \right]$$

$$+ \left[y_2 \ln y'_2 + (1-y_2) \ln (1-y'_2) \right]$$

$$= \left[\cancel{0.8 \ln (1)} + \cancel{(1-0.8) \ln (1-1)} \right]$$

$$+ \left[\cancel{-0.2 \ln (0)} + \cancel{(1-0.2) \ln (1-0)} \right]$$

$$= \left[1 \ln (0.8) + (1-0) \ln (1-0.8) \right]$$

$$+ \left[0 \times \ln (0.2) + (1-0) \ln (1-0.2) \right]$$

$$= [-0.223 + 0] + [+0 + (-0.223)] \\ = -0.446$$

for 2nd example

output: 0.6, 0.4

target: 0, 1

$$L = [0 \cdot 6 \ln(0) + (1-0.6) \ln(1-0)] \\ + [0.4 \ln(1) + (1-0.4) \ln(1-1)]$$

Prediction, $y' = 0.6, 0.4$

Actual, $y = 0, 1$

$$L = [0 \times \ln(0.6) + (1-0) \ln(1-0.6)] \\ + [1 \ln(0.4) + (1-1) \ln(1-0.4)] \\ = [0 + (-0.92)] + [1 \times (-0.92) + 0] \\ = -1.84$$

5] Control Size of Neural Network:

cost function regularization

$$J = \sum_{i=1}^N \epsilon(i) + \alpha \epsilon_p(\underline{w})$$

Hence the size minimize by J

and it depends on $\epsilon_p(\underline{w})$ which is regularization factor.

$$\begin{aligned}\epsilon_p(\underline{w}) &= \sum_{k=1}^K b(w_k)^2 \\ &= \sum_{k=1}^K \frac{w_k^2}{w_0^2 + w_k^2}\end{aligned}$$

if $\frac{w_k^2}{w_0^2 + w_k^2} < \text{threshold}$

then this weighted example
delete from ~~net~~ network.

Parameter sensitive calculation

Taylor Series :

$$f_{\delta} = \sum g_i \delta w_i + \frac{1}{2} \sum b_{ii} \delta w_i^2 + \frac{1}{2} \sum_{ij} b_{ij} \delta w_i \delta w_j$$

+ higher order class

where, $g_i = \frac{\partial f}{\partial w_i}$

$$b_{ii} = \frac{\partial^2 f}{\partial w_i^2}$$

$$b_{ij} = \frac{\partial^2 f}{\partial w_i \partial w_j}$$

for local minima/local maxima,

$$\frac{\partial f}{\partial w} = 0$$

And Saliency, $s_i = \frac{b_{ii} w_i^2}{2}$

For the effect of w_i if s_i is too less then these neuron could be removed from the network and then trained again for checking the error rate.

61

Given

$$\underline{w} = \sum_{i=1}^N \alpha_i y_i x_i \rightarrow (i)$$

$$\sum_{i=1}^N \alpha_i y_i = 0 \rightarrow (ii)$$

$$\text{And, } L(w, w_0, \alpha) = \frac{1}{2} w^T w - \sum_{i=1}^N \alpha_i [y_i (w^T x_i + w_0) - 1] \rightarrow (iii)$$

So, for optimization problem we
should maximize the Lagrangian
equation.

so, if equation (i) and (ii) set to
the equation (iii) then it makes the
dual problem which is optimized.

Dual Problem

$$L(w, w_0, \lambda) = \frac{1}{2} w^T w - \sum_{i=1}^N \lambda_i [y_i (w^T \underline{x}_i + w_0) - 1]$$

→ (i)

$$\text{As, } w = \sum_{i=1}^N \lambda_i y_i \underline{x}_i$$

$$\text{and } \sum_{i=0}^N \lambda_i y_i = 0$$

So,

$$L(w, w_0, \lambda) = \frac{1}{2} \left(\sum_{i=1}^N \lambda_i y_i \underline{x}_i \right)^T \left(\sum_{i=1}^N \lambda_i y_i \underline{x}_i \right)$$

$$- \left(\cancel{w^T \lambda_i y_i \underline{x}_i} + w_0 \sum_{i=1}^N \lambda_i y_i - \sum_{i=1}^N \lambda_i \right)$$

$$= \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N (\lambda_i \lambda_j y_i y_j \underline{x}_i^T \underline{x}_j)$$

$$- \left(\sum_{i=1}^N \left(\sum_{j=1}^N \lambda_i y_i \underline{x}_i^T \right) \lambda_i y_i \underline{x}_i \right.$$

$$\left. + \cancel{w_0 \times 0} - \sum_{i=1}^N \lambda_i \right)$$

$$= \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N (\lambda_i \lambda_j y_i y_j \underline{x}_i^T \underline{x}_j)$$

$$- \left(\sum_{i=1}^N \sum_{j=1}^N \lambda_i y_i y_j \underline{x}_i^T \underline{x}_j \right) + \sum_{i=1}^N \lambda_i$$

$$\phi = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N d_i y_i y_j \underline{x}_i^T \underline{x}_j + \sum_{i=1}^N \lambda_i$$

$$\therefore = \sum d_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N d_i y_i y_j \underline{x}_i^T \underline{x}_j$$

maximize equation

$\underline{x}_i^T \underline{x}_j$ dot product give two
matrix.

- which examples are the support vectors?

An: the examples that lie on the margin

boundary or inside the margin

boundary, which means they have

non-zero Lagrangian multiplier (α)

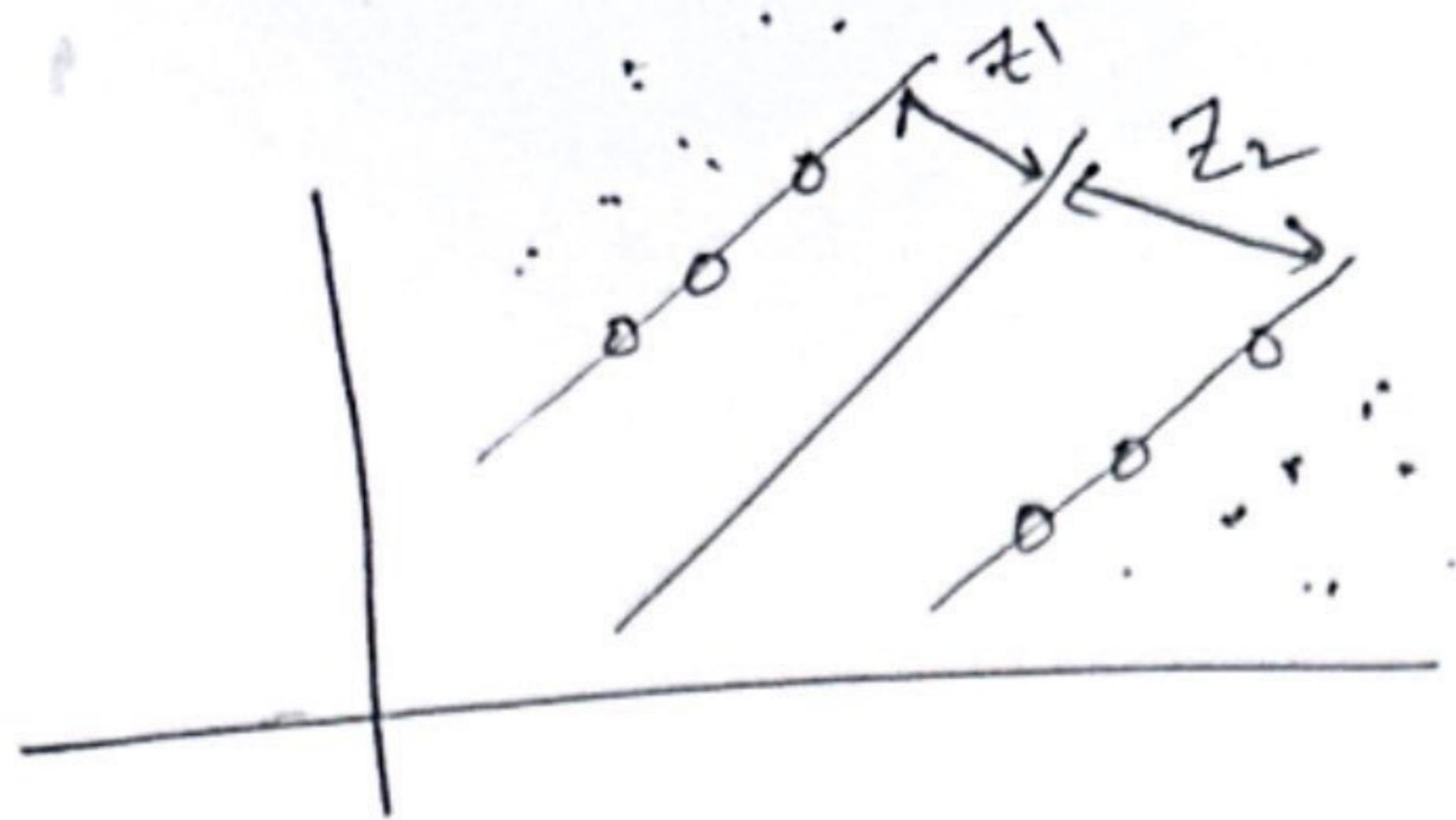
$$\therefore \alpha \neq 0$$

- for support vectors $\rightarrow \alpha \neq 0$

- for non-support vectors means

those that lie far away from the
margin boundary $\rightarrow \alpha = 0$

~~SL~~



- (i) $\underline{w}^T \underline{x}_i + w_0 \geq 1 , \quad \forall x_i \in w_1$ [lie ~~on~~ ^{z1} side]
- (ii) $\underline{w}^T \underline{x}_i + w_0 \leq -1 , \quad \forall x_i \in w_2$ [lie ^{z2} side]

from (i) = $\frac{1}{\|\underline{w}\|}$ [margin]

from (ii) = $\frac{1}{\|\underline{w}\|}$

merge these equations:

$$\frac{1}{\|\underline{w}\|} + \frac{1}{\|\underline{w}\|} = \frac{2}{\|\underline{w}\|}$$

After minimizing, for classifying
2 class by 1 equation :

$$f(w_1, w_2) = \frac{1}{2} \|\underline{w}\|^2$$

$$\text{subject to } y_i (\underline{w}^T \underline{x}_i + w_0) \geq 1$$

$$i = 1, 2, 3, \dots, N$$

And

$$\frac{\partial L}{\partial w} (w, w_0, \lambda) = 0 \rightarrow (i)$$

$$\frac{\partial L}{\partial w_0} (w, w_0, \lambda) = 0$$

$$\lambda \geq 0 \rightarrow (ii)$$

so, from
 $\Rightarrow [y_i (w^T x_i + w_0)] \geq 1$
 $\Rightarrow y_i (w^T x_i + w_0) - 1 \geq 0 \rightarrow (iv)$

from
 (ii) and (iv)
 $\lambda: [y_i (w^T x_i + w_0) - 1] \geq 0 \rightarrow (v)$
where, λ = Lagrangian multiplier
and $L(w, w_0, \lambda)$ is the Lagrangian
function

$$L(w, w_0, \lambda) = \frac{1}{2} w^T w - \sum_{i=1}^N \lambda_i y_i x_i$$

$$L(w, w_0, \lambda) = \frac{1}{2} w^T w - \sum_{i=1}^N \lambda_i [y_i (w^T x_i + w_0) - 1]$$

$$\begin{aligned} \frac{\partial}{\partial w} L(w, w_0, \lambda) &= \frac{\partial}{\partial w} \left(\frac{1}{2} w^T w - \sum_{i=1}^N (\lambda_i y_i w^T x_i + \lambda_i y_i w_0 - \lambda_i) \right) \\ &= \boxed{\sum_{i=1}^N (\lambda_i y_i x_i + \lambda_i y_i)} \end{aligned}$$

$$\Rightarrow 0 = \frac{1}{2}x^T w - \sum_{i=1}^N \alpha_i y_i x_i + b - b$$

$$\Rightarrow 0 = w - \sum_{i=1}^N \alpha_i y_i x_i$$

$$\therefore \boxed{w = \sum_{i=1}^N \alpha_i y_i x_i}$$

Again

$$\frac{d}{dw_0} L(w, w_0, d) = \frac{d}{dw_0} \left[\frac{1}{2} w^T w - \sum_{i=1}^N (\alpha_i y_i w^T x_i + \alpha_i y_i w_0 - d_i) \right]$$

$$\Rightarrow 0 = 0 - \sum_{i=1}^N (0 + \alpha_i y_i - 0)$$

$$\Rightarrow 0 = - \sum_{i=1}^N \alpha_i y_i$$

$$\therefore \boxed{\sum_{i=1}^N \alpha_i y_i = 0}$$

71

Given:

$$B = \begin{bmatrix} H & T \\ 0.4 & 0.5 \\ 0.7 & 0.3 \\ 0.5 & 0.5 \end{bmatrix}$$

$$A = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.5 & 0.2 & 0.3 \\ 0.4 & 0.2 & 0.4 \end{bmatrix}$$

$$\pi = \begin{bmatrix} 0.4 \\ 0.3 \\ 0.3 \end{bmatrix}$$

HTHH

formula: $-\ln(\pi_i, b_i, (0_i))$

And $-\ln(a_{ij}, b_j, (j))$

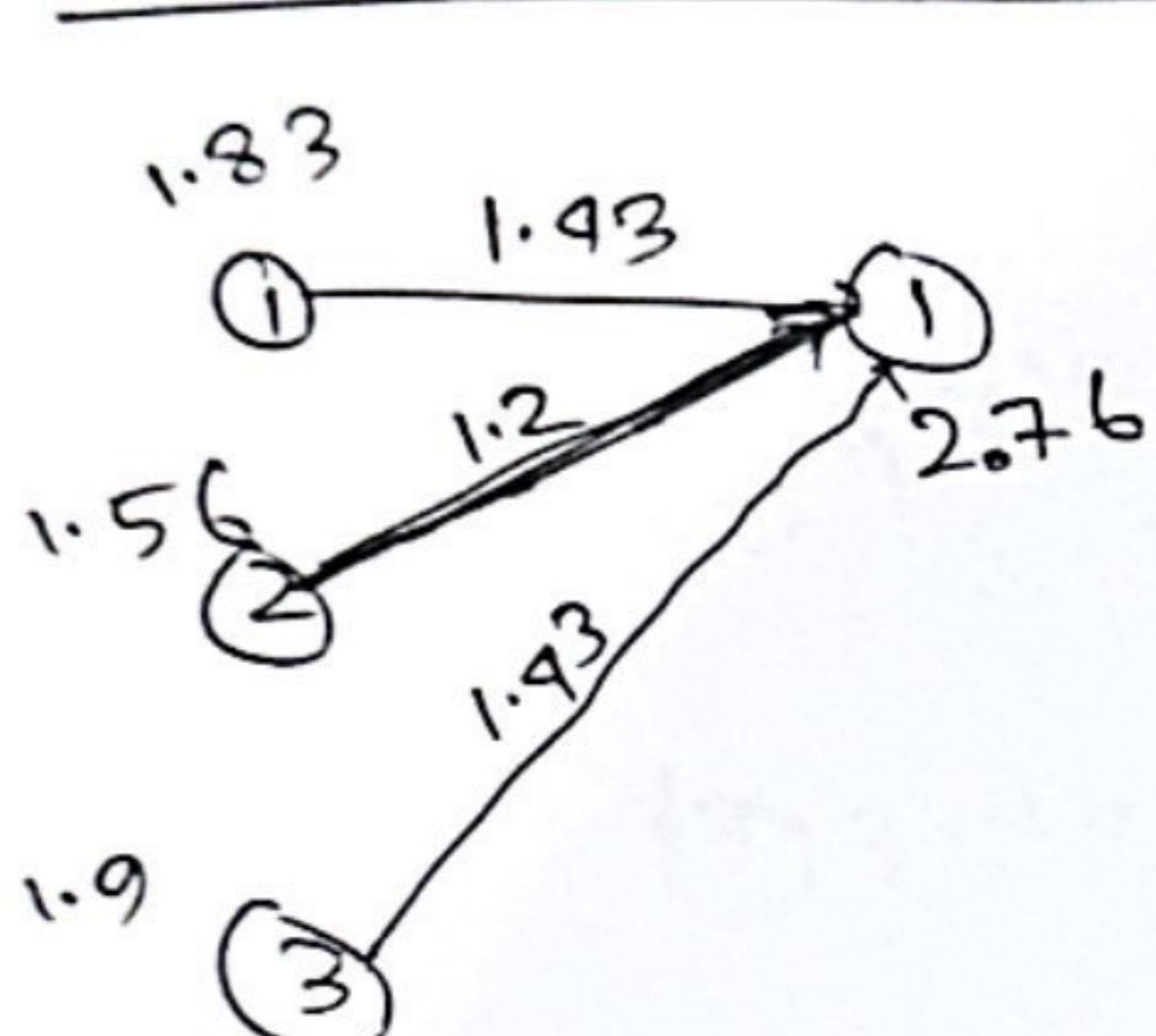
1st output: H

$$① \rightarrow -\ln(0.4 \times 0.4) = 1.83$$

$$② \rightarrow -\ln(0.3 \times 0.7) = 1.56$$

$$③ \rightarrow -\ln(0.3 \times 0.5) = 1.9$$

For 2nd output: T



$$\frac{H \quad T}{1 \rightarrow 1} -\ln(0.4 \times 0.6) = 1.23$$

$$2 \rightarrow 1 -\ln(0.5 \times 0.6) = 1.2$$

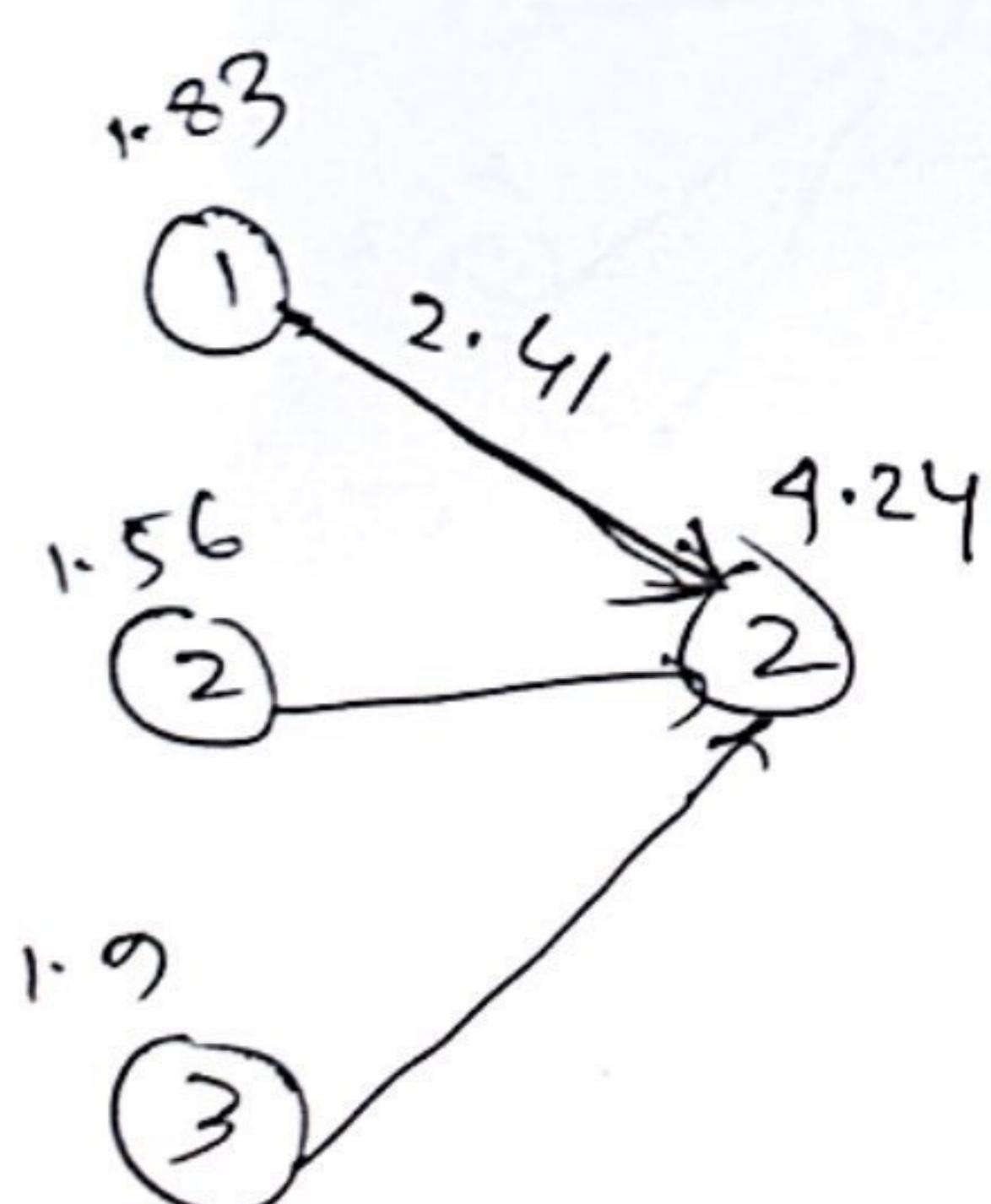
$$3 \rightarrow 1 -\ln(0.4 \times 0.6) = 1.93$$

As $2 \rightarrow 1$ is less sum

~~out probabil~~

so, $2 \rightarrow 1$ is selected

$$\text{and sum} = 1.56 + 1.2 = 2.76$$



$$\frac{H \quad T}{1 \rightarrow 2} -\ln(0.3 \times 0.3) = 2.41$$

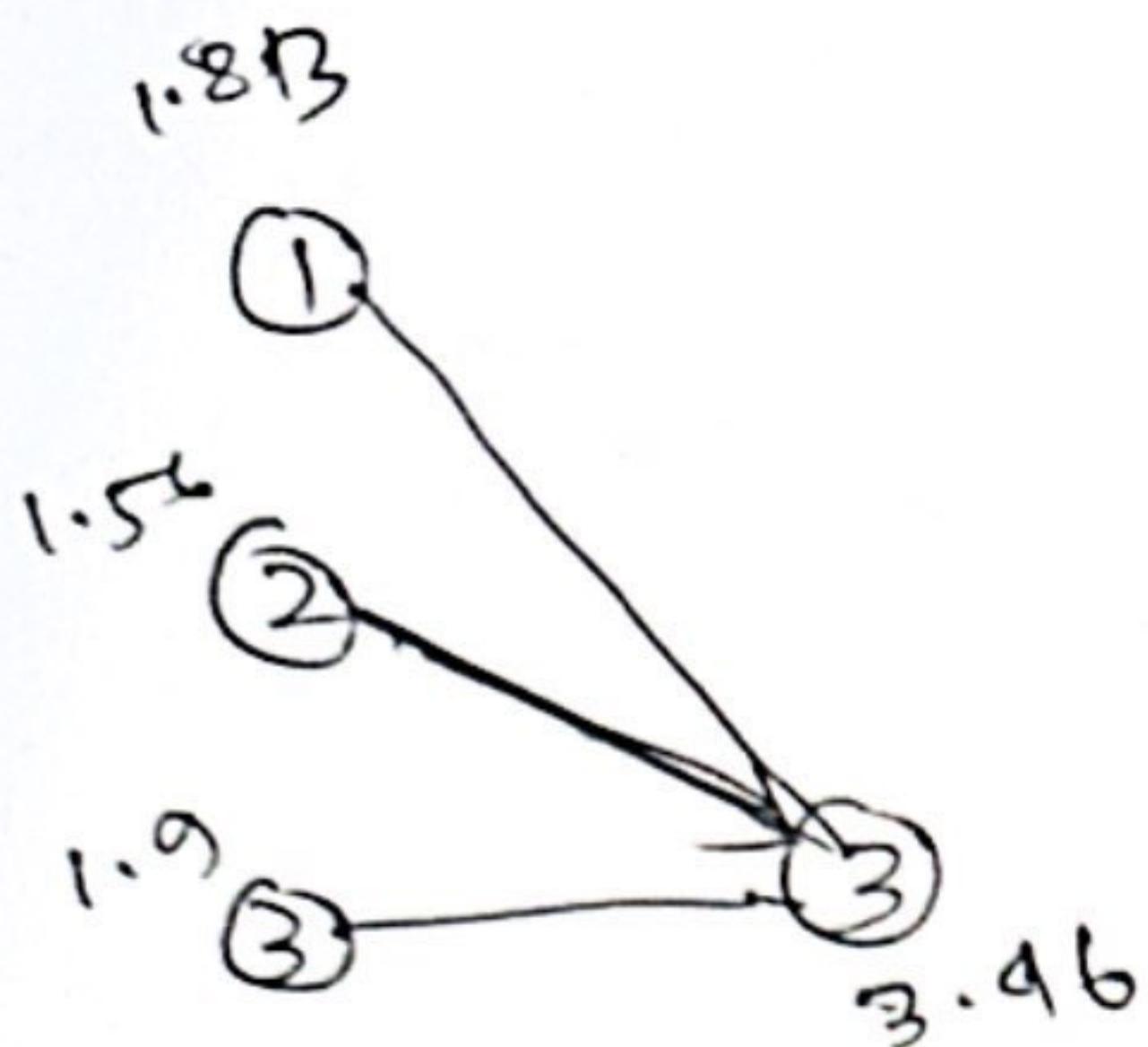
$$\text{sum} = 1.83 + 2.41 = 4.24$$

$$2 \rightarrow 2 -\ln(0.2 \times 0.3) = 2.81$$

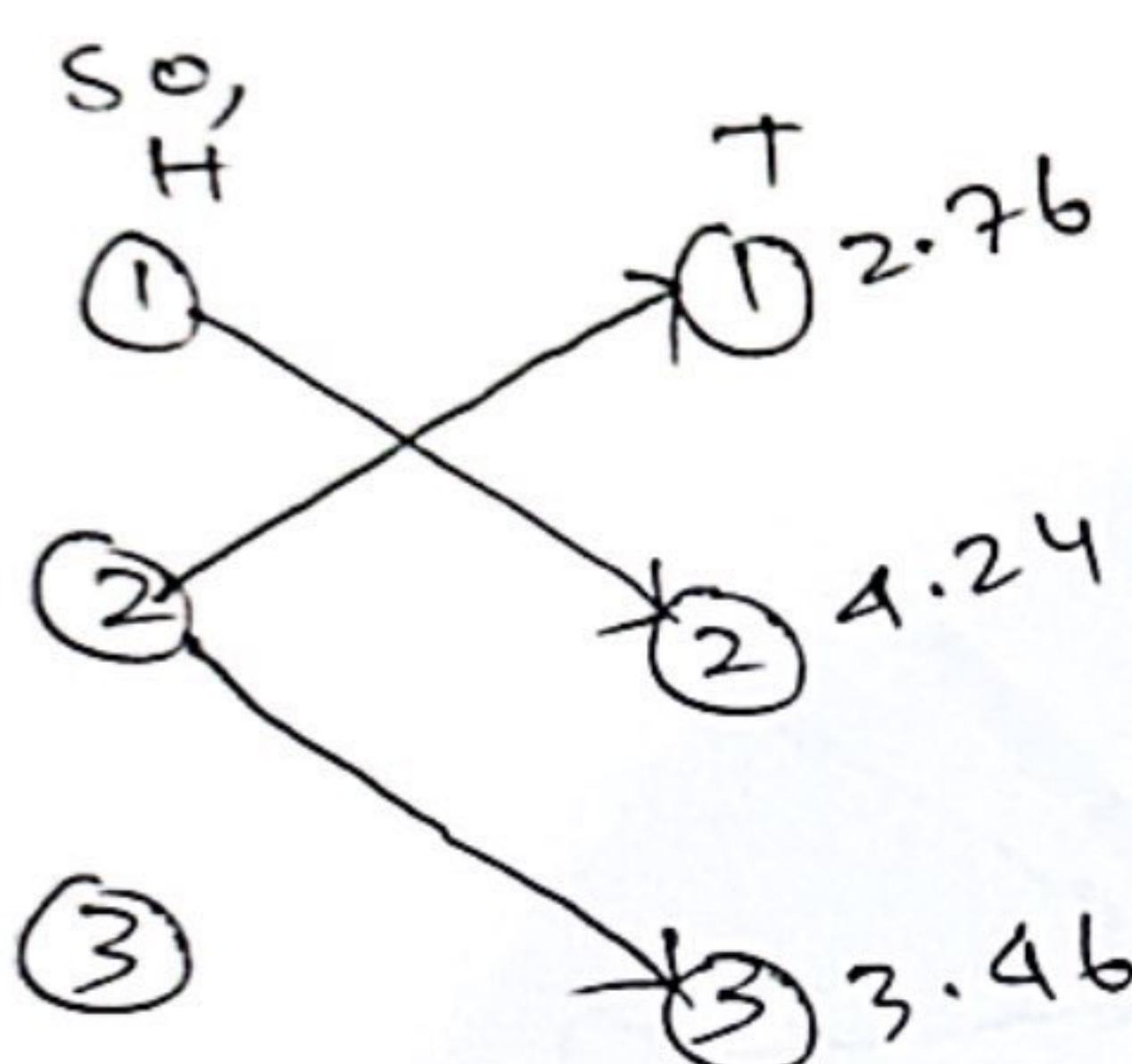
$$\text{sum} = 1.56 + 2.81 = 4.37$$

$$3 \rightarrow 2 -\ln(0.2 \times 0.3) = 2.81$$

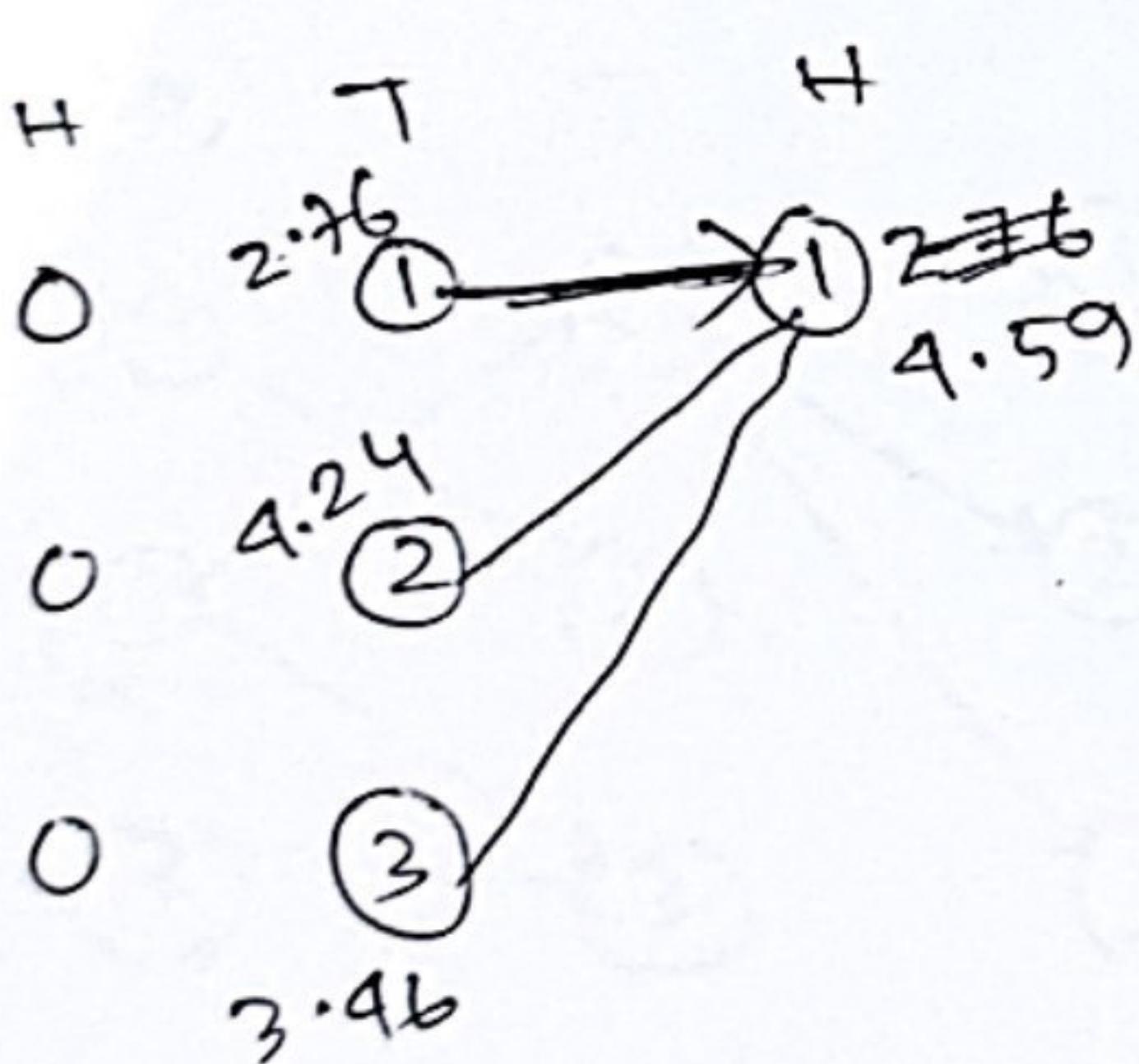
$$\text{sum} = 1.9 + 2.81 = 4.71$$



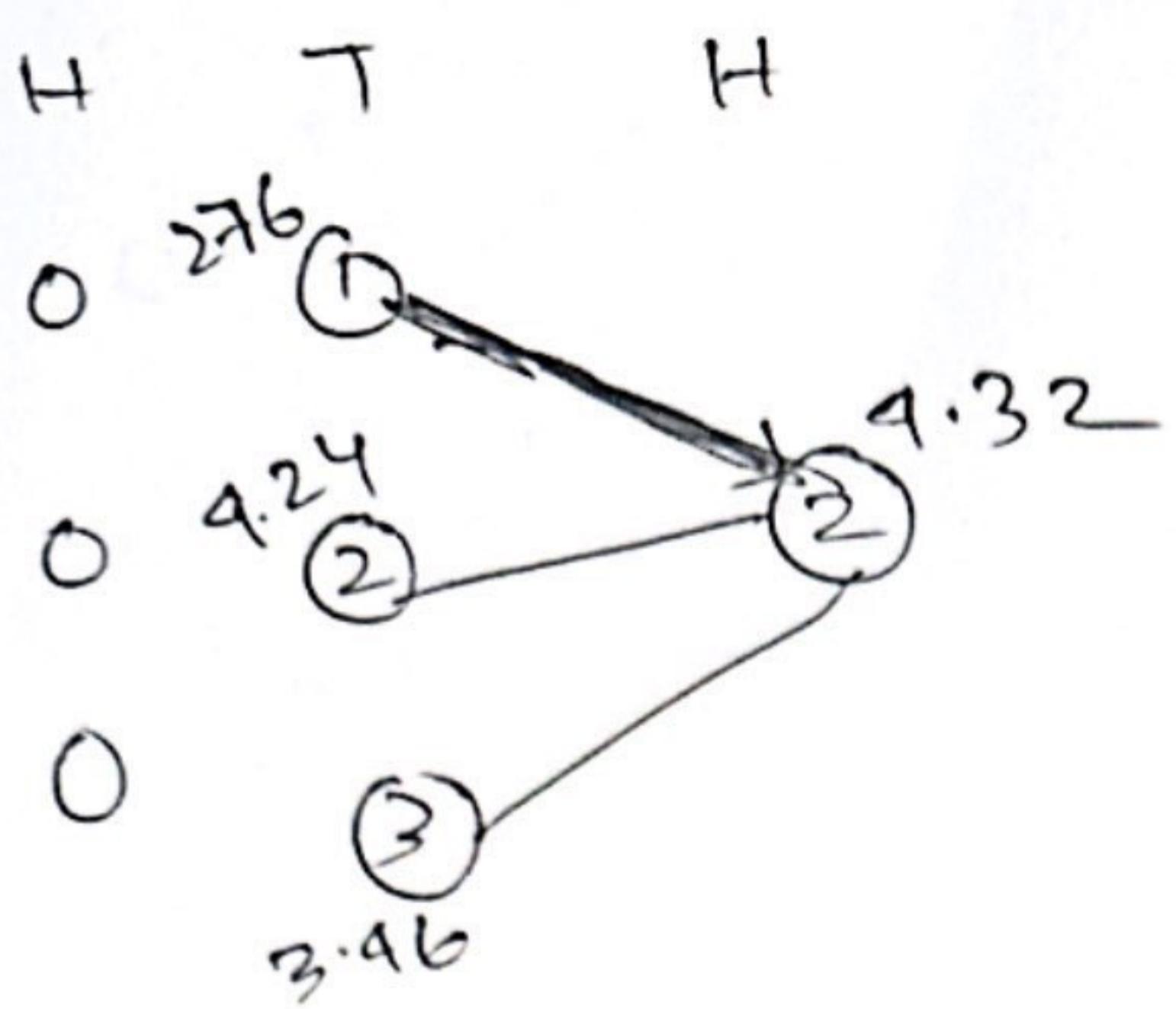
$$\begin{array}{c}
 \text{H} \quad \text{T} \\
 \hline
 1 \rightarrow 3 \\
 \xrightarrow{\text{HT}} 3 \\
 2 \rightarrow 3 \\
 \xrightarrow{\text{HT}} 3 \\
 3 \rightarrow 3
 \end{array}
 \quad -\ln(0.3 \times 0.5) = 1.9 \\
 \text{sum} = 1.83 + 1.9 = 3.73 \\
 -\ln(0.3 \times 0.5) = 1.9 \\
 \text{sum} = 1.56 + 1.9 = 3.46 \\
 -\ln(0.4 \times 0.5) = 1.61 \\
 \text{sum} = 1.9 + 1.61 = 3.51$$



For 3rd output : H



$$\begin{array}{c}
 \text{HT} \quad \text{H} \\
 \hline
 1 \rightarrow 1 \\
 2 \rightarrow 1 \\
 3 \rightarrow 1
 \end{array}
 \quad -\ln(0.4 \times 0.4) = 1.83 \\
 \text{sum} = 2.76 + 1.83 = 4.59 \\
 -\ln(0.5 \times 0.9) = 1.61 \\
 \text{sum} = 4.24 + 1.61 = 5.85 \\
 -\ln(0.4 \times 0.9) = 1.83 \\
 \text{sum} = 3.46 + 1.83 = 5.29$$



$$H \xrightarrow{1} T \xrightarrow{2} H$$

$$-\ln(0.3 \times 0.7) = 1.56$$

$$\text{sum} = 2.76 + 1.56 = \underline{4.32}$$

$$H \xrightarrow{2} T \xrightarrow{2} H$$

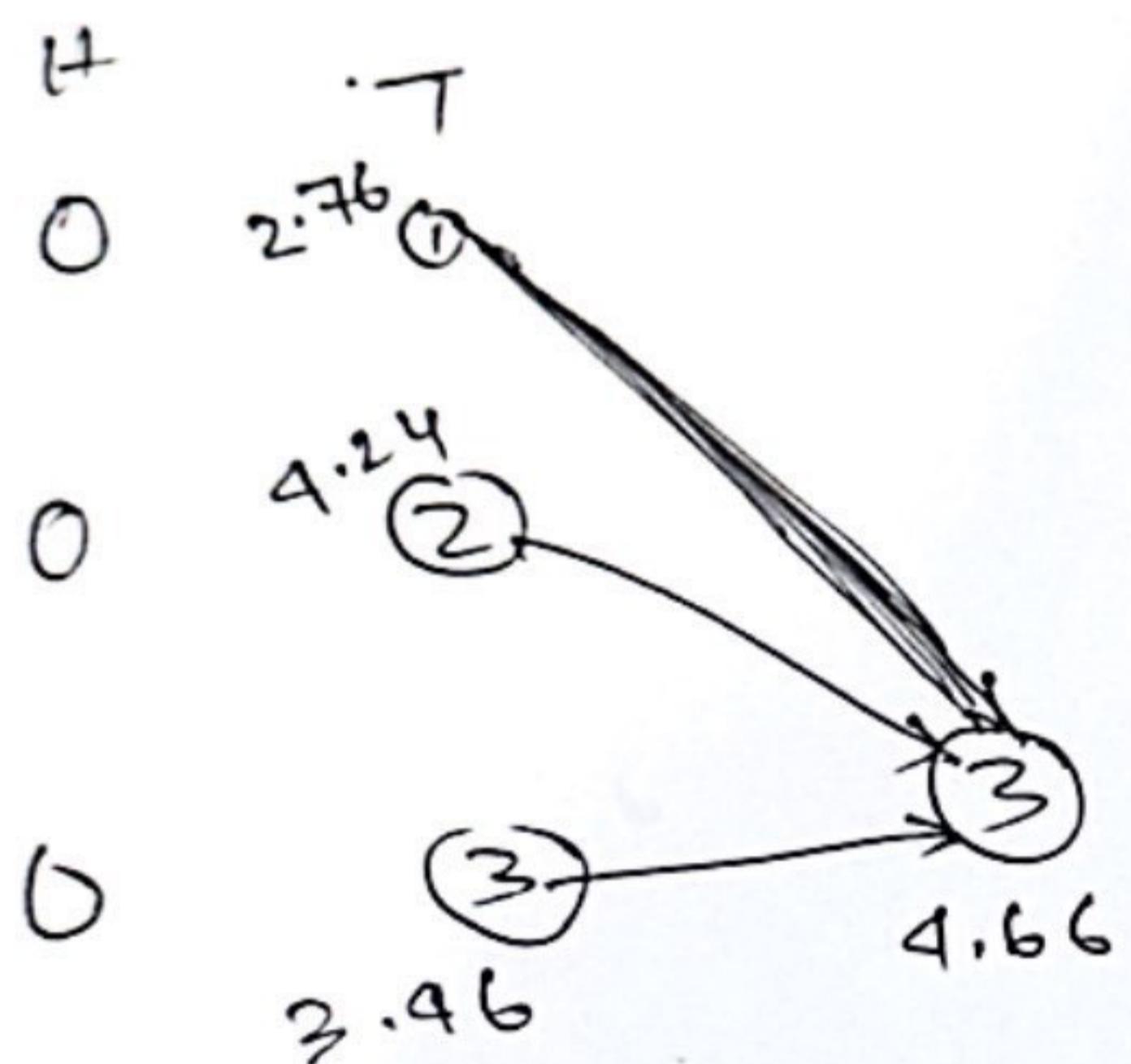
$$-\ln(0.2 \times 0.7) = 1.97$$

$$\text{sum} = 4.24 + 1.97 = \underline{6.21}$$

$$H \xrightarrow{3} T \xrightarrow{2} H$$

$$-\ln(0.2 \times 0.7) = 1.97$$

$$\text{sum} = 3.96 + 1.97 = \underline{5.93}$$



$$H \xrightarrow{1} T \xrightarrow{3} H$$

$$-\ln(0.3 \times 0.5) = 1.9$$

$$\text{sum} = 2.76 + 1.9 = \underline{4.66}$$

$$H \xrightarrow{2} T \xrightarrow{3} H$$

$$-\ln(0.3 \times 0.5) = 1.9$$

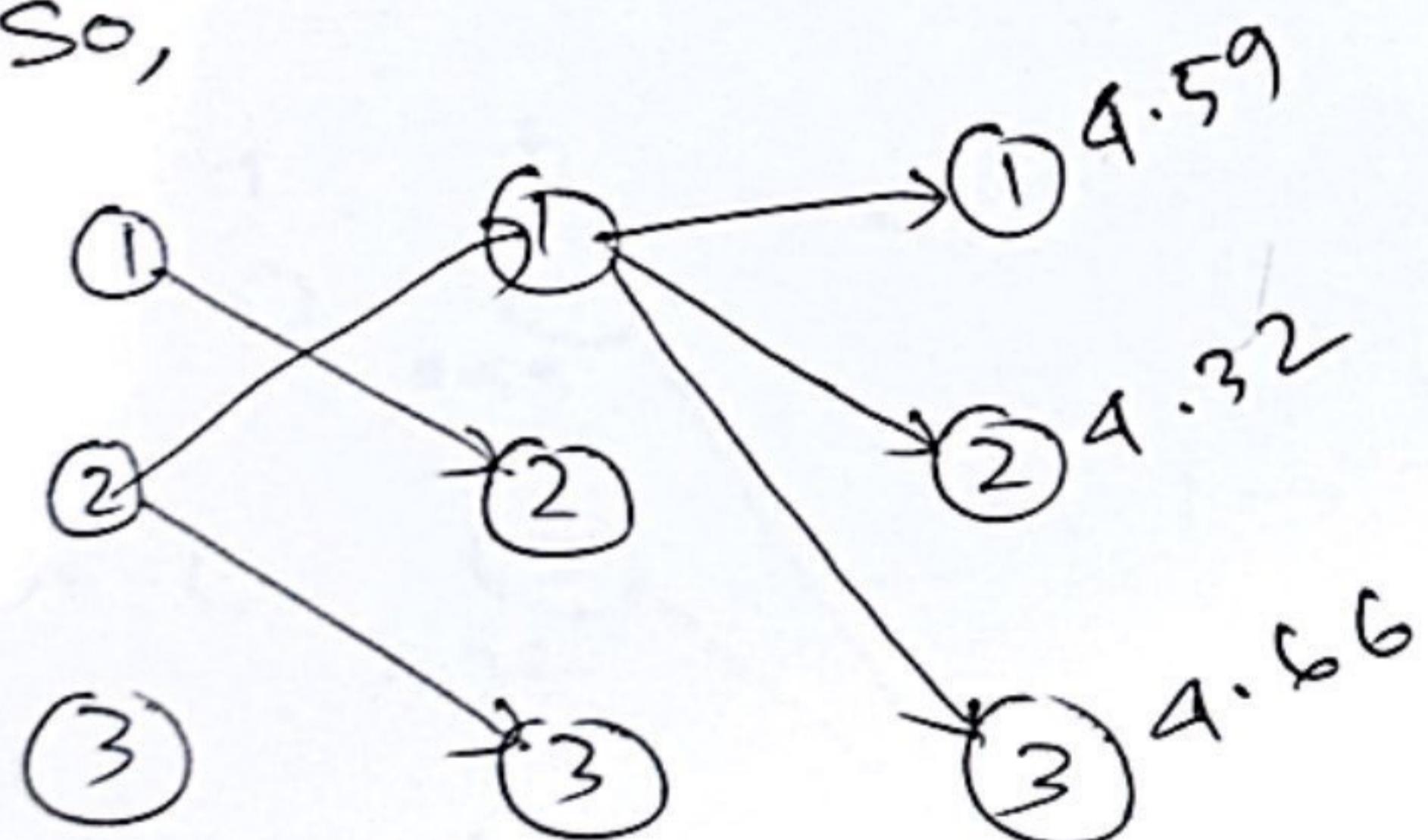
$$\text{sum} = 4.24 + 1.9 = \underline{6.14}$$

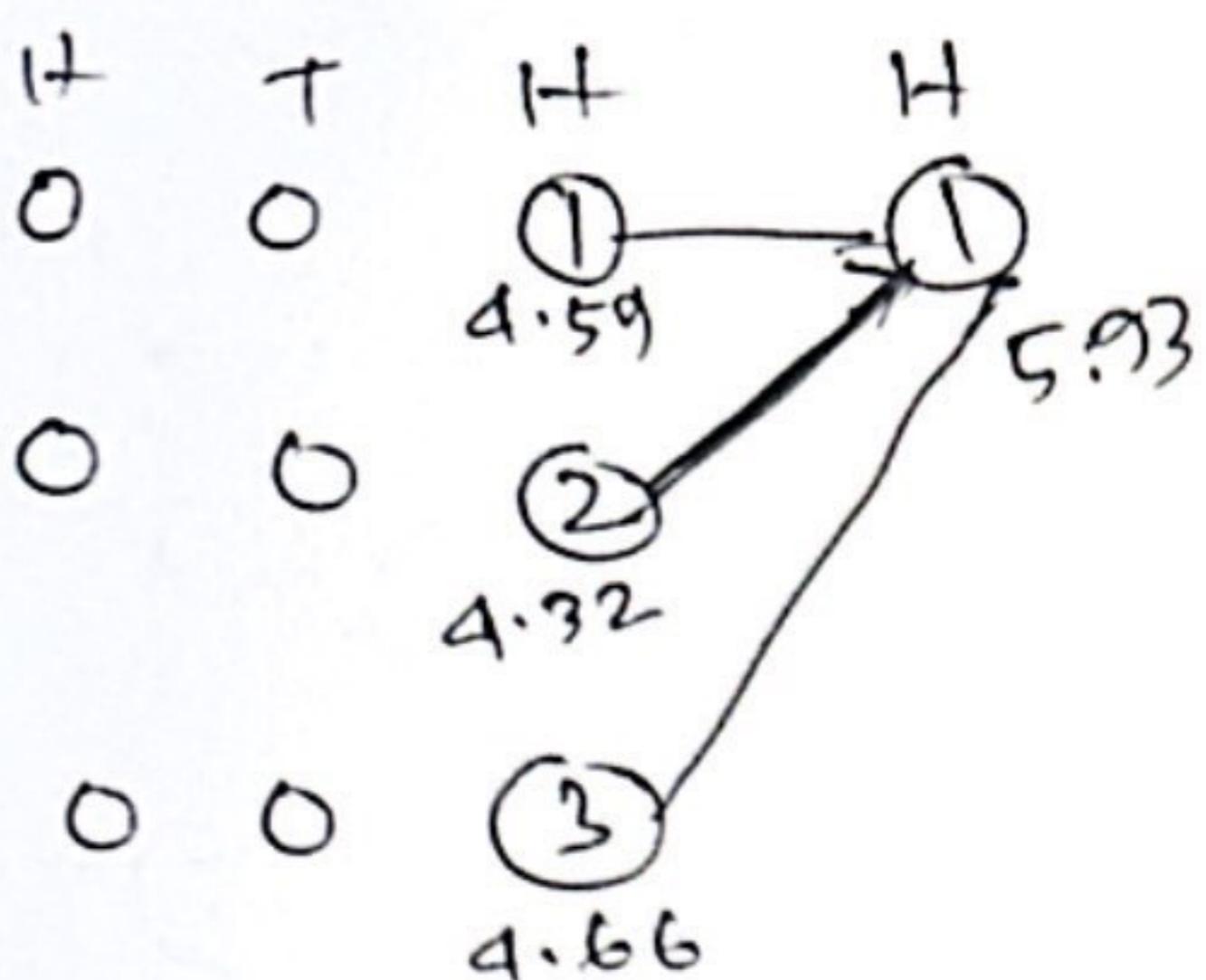
$$H \xrightarrow{3} T \xrightarrow{3} H$$

$$-\ln(0.4 \times 0.5) = 1.61$$

$$\text{sum} = 3.96 + 1.61 = \underline{5.57}$$

So,





$$H \xrightarrow{T} 1 \rightarrow H - \ln(0.9 \times 0.4) = 1.83$$

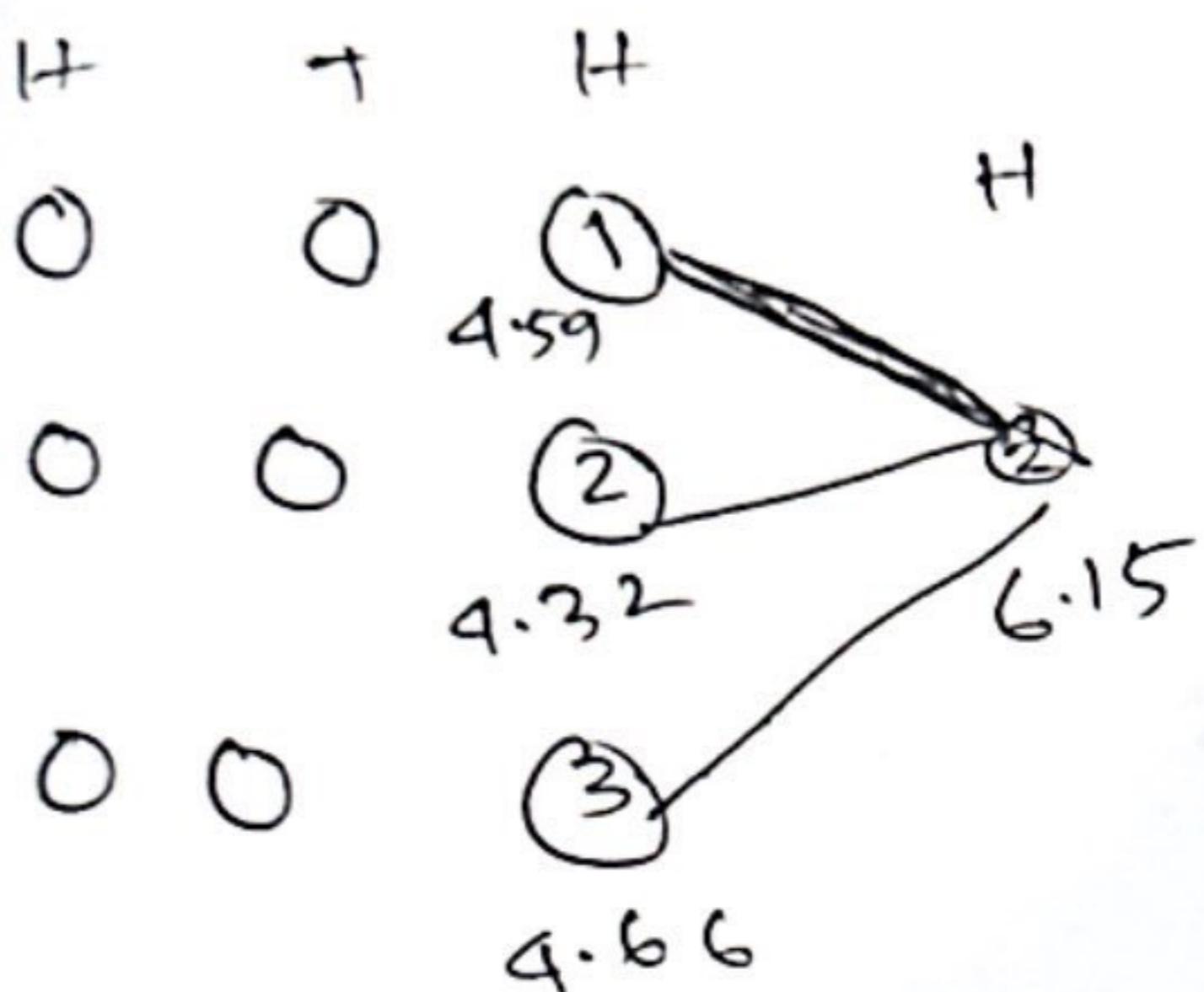
$$\text{sum} = 4.59 + 1.83 = 6.42$$

$$2 \rightarrow 1 - \ln(0.5 \times 0.4) = 1.61$$

$$\text{sum} = 4.32 + 1.61 = \underline{\underline{5.93}}$$

$$3 \rightarrow 1 - \ln(0.4 \times 0.4) = 1.83$$

$$\text{sum} = 4.66 + 1.83 = 6.49$$



$$1 \rightarrow 2 - \ln(0.3 \times 0.7) = 1.56$$

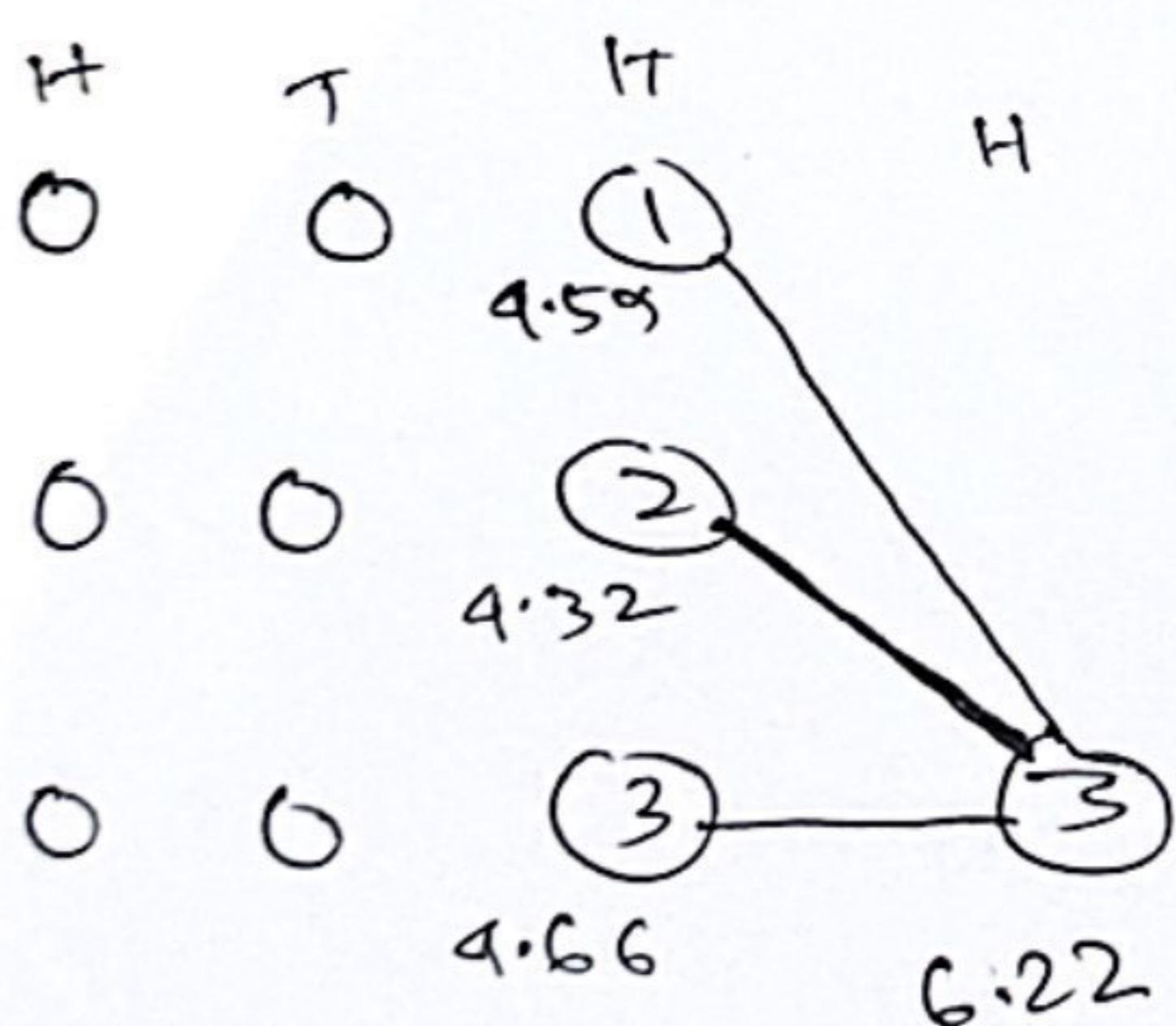
$$\text{sum} = 4.59 + 1.56 = \underline{\underline{6.15}}$$

$$2 \rightarrow 2 - \ln(0.2 \times 0.7) = 1.97$$

$$\text{sum} = 4.32 + 1.97 = 6.29$$

$$3 \rightarrow 2 - \ln(0.2 \times 0.7) = 1.97$$

$$\text{sum} = 4.66 + 1.97 = 6.63$$



$$1 \rightarrow 3 - \ln(0.3 \times 0.5) = 1.9$$

$$\text{sum} = 4.59 + 1.9 = \underline{\underline{6.49}}$$

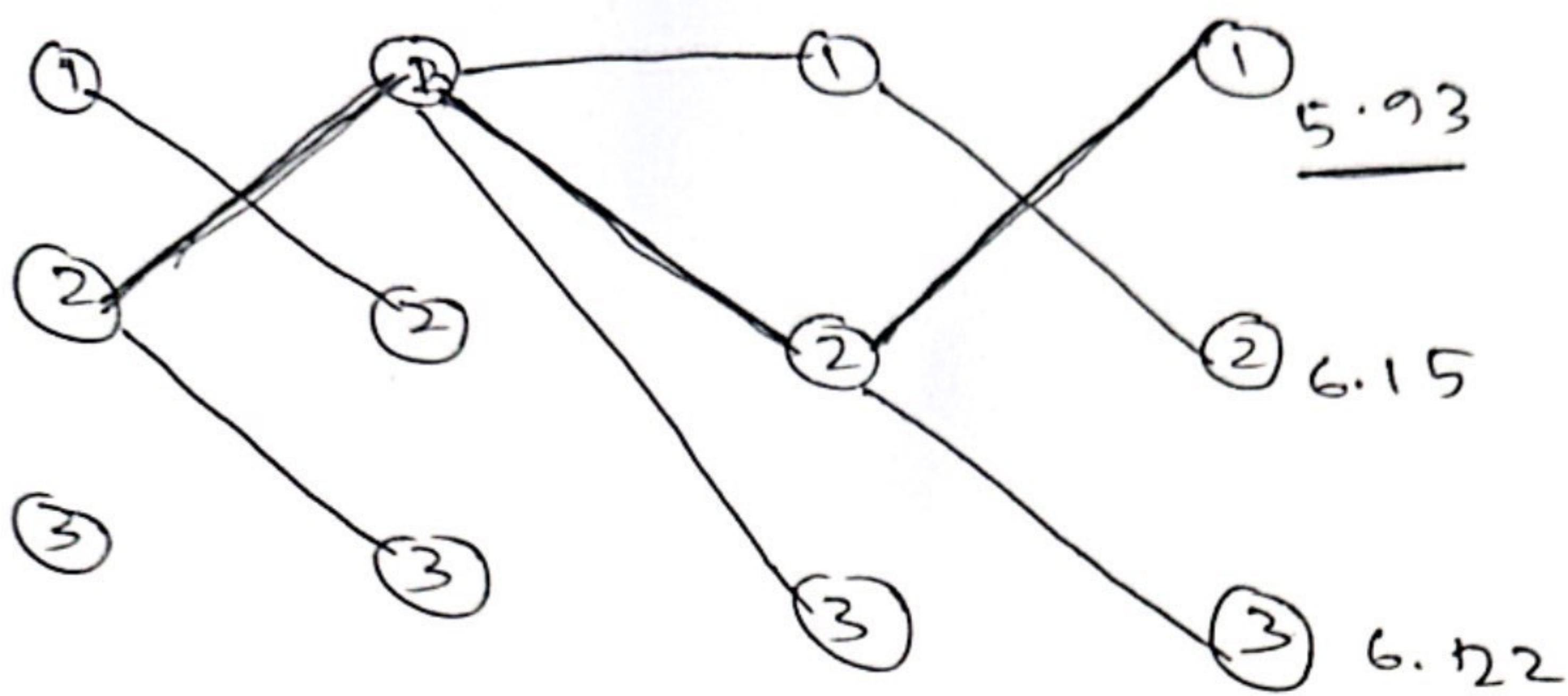
$$2 \rightarrow 3 - \ln(0.3 \times 0.5) = 1.9$$

$$\text{sum} = 4.32 + 1.9 = \underline{\underline{6.22}}$$

$$3 \rightarrow 3 - \ln(0.9 \times 0.5) = 1.61$$

$$\text{sum} = 4.66 + 1.61 = \underline{\underline{6.27}}$$

So Final route



So the route or sequence is

$$= 2 \rightarrow 1 \rightarrow 2 \rightarrow 1$$

$$\begin{aligned} \text{So, probability} &= e^{-5.93} \\ &= 2.66 \times 10^{-3} \end{aligned}$$

Probability of ~~match~~ HTHH

$$\alpha_t(i) = P(o_1, o_2, o_3, \dots, o_t, i_t=i)$$

H

$$\alpha_1(i) = \pi_i b_i(o_1) \quad \therefore$$

$$\therefore \alpha_1(1) = 0.4 \times 0.4 = 0.16 \\ (\pi_1) \times (b_1)$$

$$\alpha_1(2) = \pi_2 \times b_2(o_1) = 0.3 \times 0.7 = 0.21$$

$$\alpha_1(3) = \pi_3 \times b_3(o_1) = 0.3 \times 0.5 = 0.15$$

Ans $\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1})$

I

$$\therefore \alpha_2(1) = [\alpha_1(1) \times a_{11} + \alpha_1(2) \times a_{21} + \alpha_1(3) \times a_{31}] \\ \times b_1(o_2)$$

$$= [0.16 \times 0.4 + 0.21 \times 0.5 + 0.15 \times 0.9]$$

$$\times 0.6$$

$$= 0.1374$$

$$\alpha_2(2) = [\alpha_1(1) \times a_{12} + \alpha_1(2) \times a_{22} + \alpha_1(3) \times a_{32}] b_2(o_2)$$

$$= (0.16 \times 0.3 + 0.21 \times 0.2 + 0.15 \times 0.2) \times 0.3$$

$$= 0.036$$

$$\alpha_2(3) = [\alpha_1(1) \times a_{13} + \alpha_1(2) \times a_{23} + \alpha_1(3) \times a_{33}] b_3(o_2)$$

$$= (0.16 \times 0.3 + 0.21 \times 0.3 + 0.15 \times 0.4) \times 0.5$$

$$= 0.0955$$

$$\overline{\alpha}_3(1) = \left[\alpha_2(1) \times a_{11} + \alpha_2(2) \times a_{21} + \alpha_2(3) \times a_{31} \right] \times b_1(0_3)$$

$$= \left[0.1374 \times 0.4 + 0.036 \times 0.5 + 0.0855 \times 0.9 \right] \times 0.9$$

$$= 0.043$$

$$\overline{\alpha}_3(2) = \left[\alpha_2(1) \times a_{12} + \alpha_2(2) \times a_{22} + \alpha_2(3) \times a_{32} \right] \times b_2(0_3)$$

$$= \left[0.1374 \times 0.3 + 0.036 \times 0.2 + 0.0855 \times 0.2 \right] \times 0.7$$

$$= 0.046$$

$$\overline{\alpha}_3(3) = \left[\alpha_2(1) \times a_{13} + \alpha_2(2) \times a_{23} + \alpha_2(3) \times a_{33} \right] \times b_3(0_3)$$

$$= \left[0.1374 \times 0.3 + 0.036 \times 0.3 + 0.0855 \times 0.9 \right] \times 0.5$$

$$= 0.043$$

$$\overline{\alpha}_4(1) = \left[\alpha_3(1) \times a_{11} + \alpha_3(2) \times a_{21} + \alpha_3(3) \times a_{31} \right] \times b_1(0_4)$$

$$= \left[0.043 \times 0.4 + 0.096 \times 0.5 + 0.043 \times 0.9 \right] \times 0.4$$

$$= 0.023$$

$$\overline{\alpha}_4(2) = \left[\alpha_3(1) \times a_{12} + \alpha_3(2) \times a_{22} + \alpha_3(3) \times a_{32} \right] \times b_2(0_4)$$

$$= \left[0.043 \times 0.3 + 0.096 \times 0.2 + 0.043 \times 0.2 \right] \times 0.7$$

$$= 0.0215$$

$$\overline{\alpha}_4(3) = \left[0.043 \times 0.3 + 0.096 \times 0.3 + 0.043 \times 0.9 \right] \times 0.5$$

$$= 0.022$$

$$\begin{aligned} P(HTHH) &= \alpha_4(1) + \alpha_4(2) + \alpha_4(3) \\ &= 0.023 + 0.0215 + 0.022 \\ &= 0.0665 \end{aligned}$$

8 / Previous CT-3 /

Phase - 1

Input size = $227 \times 227 \times 3$

Number of filters = 100

filter size = ~~21~~ $21 \times 21 \times 3$

stride = 9

zero padding = 1

$$\begin{aligned} \text{output size} &= \frac{w - f + 2p}{s} + 1 \\ &= \frac{227 - 21 + 2 \times 1}{9} + 1 \\ &= 53 \end{aligned}$$

\therefore After 1st phase output of the convolution layer is = $53 \times 53 \times 100$

For 2nd phase

Input size = $53 \times 53 \times 100$

Number of filters = 50

filter size = $7 \times 7 \times 100$

stride = 2

zero padding = 0

$$\text{output size} = \frac{53 - 7 + 2 \times 0}{2} + 1 \\ = 29$$

\therefore convolution Layer = $29 \times 29 \times 50$

For 3rd phase

Input size = $29 \times 29 \times 50$

filters = 10

filter size = $5 \times 5 \times 50$

stride = 1

zero padding = 0

$$\text{output size} = \frac{29 - 5 + 2 \times 0}{1} + 1 \\ = 20$$

convolution Layer = $20 \times 20 \times 10$

Total number of neuron =

$$(53 \times 53 \times 100) + (29 \times 29 \times 50) + (20 \times 20 \times 10) \\ = 313700 (\text{Ans})$$

Q1 Implementation of Matrix Multiplication out 1st phase

output : $53 \times 53 \times 100$

filter size was : $21 \times 21 \times 3$

So, First matrix,

(1323×2809)

$$A = \begin{bmatrix} (1,1) & (1,2) & (1,3) & \dots & (1,2809) \\ (2,1) & - & - & - & - \\ \vdots & \vdots & \vdots & - & - \\ (1323,1) & - & - & - & - \end{bmatrix}$$

For 2nd matrix (100×1323)

$$B = \begin{bmatrix} (1,1) & (1,2) & (1,3) & \dots & (1,1323) \\ (2,1) & - & - & - & - \\ \vdots & \vdots & \vdots & - & - \\ (100,1) & - & - & - & - \end{bmatrix}$$

$$\therefore A \times B = \begin{bmatrix} (1,1) & (1,2) & \dots & (1,2809) \\ (2,1) & - & - & - \\ \vdots & \vdots & \vdots & - \\ (100,1) & - & - & - \end{bmatrix}^{100 \times 2809}$$

91 (2nd Half)

If we want to avoid pooling for downsizing the output of a convolutional layer, a common alternative is to use a convolutional layer with a larger stride.

$$\text{output size} = \frac{w-f+2p}{\text{stride}} + 1$$

$$\therefore \text{output size} \propto \frac{1}{\text{stride}}$$

so, by increasing stride we can downsizing the output.

10

Forwarded Method

$$\alpha_t(i) = P(o_1, o_2, o_3, \dots, o_T | s)$$

$$\text{for } \alpha_1(1) = \pi_1 b_1(o_1) \quad \begin{cases} 1 \text{ observed,} \\ \text{state} = 1 \end{cases}$$

$$\alpha_1(2) = \pi_2 b_2(o_1) \quad \begin{cases} 1, 2 \text{ observed} \\ \text{state} = 2 \end{cases}$$

$$\alpha_2(2) = \pi_2 b_2(o_2) \quad \begin{cases} 1, 2 \text{ observed} \\ \text{state} = 2 \end{cases}$$

$$\therefore \alpha_1(i) = \pi_i b_i(o_1) \quad \begin{matrix} 1 \leq i \leq N \\ \longrightarrow(i) \end{matrix}$$

for $t = 1, 2, 3, 4$

$$\sum_{i=1}^4 \alpha_t(i) = \alpha_4(1) + \alpha_4(2) + \alpha_4(3) \\ + \alpha_4(4)$$

where α .

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1}) \quad \boxed{1 \leq j \leq N}$$

$$\therefore P(O|A) \in \cancel{\sum_{i=1}^N \alpha_t(i)}$$

$$P(A, B) = P(A|B) P(B)$$

AS

$$\alpha_2(j) = P(O_1, O_2, i_2=j)$$

$$= \sum P(O_2, i_2=j | O_1, \text{from state } i)$$

$$P(O_2 \text{ from state } i)$$

$$= \sum P(O_2 | i_2=j, O_1 \text{ from state } i)$$

$$P(i_2=j | O_1 \text{ from state } i) P(O_1 | i=j)$$

$$P(i_1=i)$$

$$= \sum [P(O_2 | i_2=j) (P(i_2=j | i_1=i))]$$

$$[P(O_1 | i_1=i) P(i_1=i)]$$

$$= \sum b_j(O_2) a_{ij} \times b_i(O_1) \pi_i$$

$$= [\sum \pi_i b_i(O_1) a_{ij}] b_j(O_2)$$

$$= [\sum a_i(i) a_{ij}] b_j(O_2)$$

Complexity : $(N+1) \times (T-1) \times N + N$

$$\approx N^2 T \quad \text{if } T = 1000 \\ N = 5$$

$$\approx 3000$$

Hidden Markov Model

Calculate the probability of a sequence of observation :

$$P(O_1, O_2, O_3, O_4, \dots, O_T) = ?$$

$$\text{means } P(O_1, O_2, O_3, \dots, O_T | \lambda) = ?$$

$$\Rightarrow P(O | \lambda) = ?$$

Ans: Let, $I = I_1, I_2, I_3, \dots, I_T$

we know,

$$P(O | I, \lambda) = b_{i_1}(O_1) \times b_{i_2}(O_2) \times \dots \times b_{i_T}(O_T)$$

$$P(I | \lambda) = (\pi_{i_1} a_{i_1 i_2}) \times \dots$$

$$= \pi_{i_1} a_{i_1 i_2} a_{i_2 i_3} a_{i_3 i_4} \dots a_{i_{T-1} i_T}$$

$$\therefore P(O | \lambda) = \sum P(O | I, \lambda) P(I | \lambda)$$

$$= \sum \pi_{i_1} b_{i_1}(O_1) a_{i_1 i_2} b_{i_2}(O_2) a_{i_2 i_3} b_{i_3}(O_3) \dots a_{i_{T-1} i_T} b_{i_T}(O_T)$$

Complexity :

$$P(O | I, \lambda) = T$$

$$P(I|\lambda) = T$$

Total states = N

$$(T+T) \times N^T$$
$$= 2T N^T$$

$$\text{int } T=100, N=5 \approx 10^{75}$$

$$[N^1 \times N^2 \times N^3 \times N^4 \dots]$$

forward method is better for
Markov Model